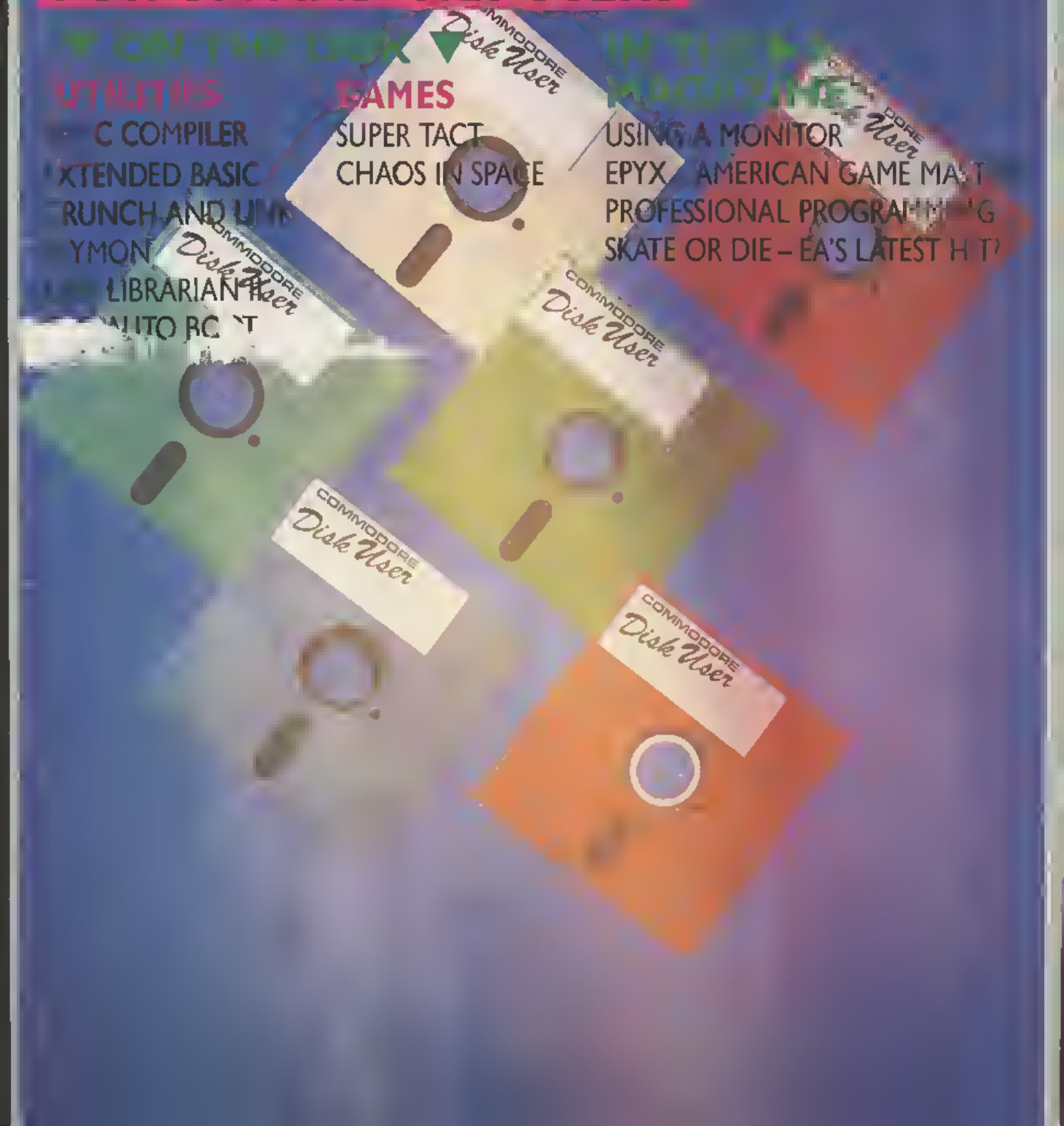
MARCH/APRIL 1988
£2.50

COMMODORE Disk User

FOR C64 AND C128 USERS



SUBSCRIPTIONS SENSATIONS!

AT COVER PRICE ONLY!

Any of the magazines from the list below can be delivered to your door with : NO handling charge NO postage costs

A&B Computing	£18.00
Aeromodeller	£22.20
Citizens Band	£16.20
Clocks	£27.00
Commodore Disk User	£15.00
Disk User	£17.70
Electronics Digest	£11.00
Electronics Today International	£16.80
Ham Radio Today	£15.60
Military Modelling	£15.60
Model Boats	£15.60
Model Engineer	£26.40
Photography	£15.00
Photoplay	£12.00
Popular Crafts	£16.20
Radio Control Model Cars	£16.20
Radio Control Boat Modeller	£7.50
RCM&E	£14.40
Radio Control Scale Aircraft Quarterly	£9.00
Radio Modeller	£14.40
Sea Classic International	£11.70
Scale Models International	£15.00
Video Today	£15.00
Which Video?	£15.00
Woodworker	£15.60
Your Commodore	£14.40
Model Railways	£13.20
Practical Wargamer	£7.80

The Above Prices Refer to U.K. Subscriptions Only

This offer is also open to existing U.K. subscribers who may wish to extend their current subscriptions.

Please commence my subscription(s) To ..... with the ..... Issue
I enclose my cheque/money order for £ ..... made payable to Argus Specialist Publications Ltd.
or debit £ ..... from my Access/Barclaycard No. ..... to ..... Signature .....
valid from .....
Name .....
Address .....

Send this form with your remittance to:
INFONET LTD. (N.R.S. 88) 5 River Park Estate, Berkhamsted, Herts. HP4 1HL.

CONTENTS

Commodore Disk User
Volume 1 Number 3
March/April 1988

IN THE MAGAZINE

Disk Info 4
How to use this month's disk

Update 4
All the latest news from the world of Commodore disk computing

Reviews 7
The latest software releases - and what we think of them

Software over the Rainbow 14
Norman Doyle tells you how to get a bit of local colour into your programs

Contributions 26
How to write for Commodore Disk User

Level-headed thoughts 31
How to create a megagame

Monitoring monitors 33
What are they for, and which one's best?

Disk Dungeons 35
Adventuring advice and opinion

Epyx 38
A look at the best-selling US software house

Professional programming 40
Make your Basic programs look better - and run better

ON THE DISK

Super-tact 13
Tactics are of the essence in this game

Chaos in Space 13
A shoot-em-up that's deceptively different

C-2ap 16
Speed is the name of the game with this superb compiler

Basic+ 20
A comprehensive Basic extension

Tape archive 21
Be safe and back your disks up to tape

Link and Crunch 22
Running out of memory/disk space? Not any more

Psymon 24
A full-facility machine-code monitor

Disk Librarian II 27
An updated version of our powerful disk catalogue utility

128 Autoboot 30
For 128 owners - load C64 programs at C128 speed

Epyx baseball

Editor: STUART COOKE
Deputy Editor: FIN FAHEY
Designer: KATHY HOWES
Advertisement Manager: STUART TAYLOR
Copy Control: LAURA CHAMPION
Origination: EBONY TYPESETTING

Distribution: S.M DISTRIBUTION
Printed by: CHASE WEB, PLYMOUTH

ARGUS PRESS GROUP

Commodore Disk User is a bi-monthly magazine published on the 3rd Friday of every alternate month. Argus Specialist Publications Limited, Commodore Disk user 1 Golden Square London, W1R 3AB Telephone 01 437 0626 Telex 8811896

UPDATE

Update

All the latest from the wonderful world of Commodore disk computing

Back to the 'Nam

The Vietnam War revival continues in computer games form with the release of a new package from Cascade Games. Titled '19 Part 1 - Boot Camp', the game follows the progress of a 19-year old American draftee from basic training through to the war itself.

In order to produce the game, Cascade researched into techniques of army basic training by sending two of their directors, John Lewis and Nigel Stevens on an army assault course on the Yorkshire moors (see pic). The rest of the company must be hoping that this does not presage radical changes in management techniques.

19 Part 1 is based on the Paul Hardcastle hit about Vietnam, and Cascade is billing it as 'the ultimate simulation of true warfare' - an ambitious hope. The game consists of sequences of arcade action with a sound track based on the original single.

The company's next release will be '19 Part 2 - Vietnam', and the score obtained on Part 1 of the program can be retained for Part 2.

19 Part 1 will come with a detailed full colour map of the playing area and will be available on disk for £14.95. For further information contact Cascade Games on (0423) 523325.

Now there's an Alternative

Many well-known computer games from the past are being being released at budget prices by Alternative Software. Alternative has obtained the rights to a number of products from Piranha, Audiogenic, Incentive and Bubble Bus.

The Piranha Games, many of which were very popular include Trap Door, Popeye, Rogue Trooper, Strike Force Cobra and Nosferatu - the first two are already available and the rest will be following soon. All games will be released at £1.99 apiece. The company is also releasing Incentive's Mooncresta and Psycastria from Audiogenic.

Cascade hit the assault course

Alternative was formed less than a year ago and has proved very successful at marketing chart software. At the moment the company is marketing cassette-based games only. For details contact Alternative on (0977) 797777.

DISK INSTRUCTIONS

Before you use your disk for the first time, read this

We have done our best to make sure that Commodore Disk User will be compatible with all versions of the C64 and C128 computers and their associated disk drives.

Getting the programs up and running should not present you with any difficulties at all, simply put your disk in the drive and enter the following command:

LOAD "MENU",8,1

Once the disk menu has loaded you will be able to start any of the programs simply by pressing the letter that is to the left of the program that you want to use.

C128 users please note that you should be in C64 mode when using the disk. You can enter C64 mode by either:

i) Holding down the Commodore key (bottom left of the keyboard) when turning the computer on or,

ii) After turning the computer on type GO64 and answer "Y" when prompted "ARE YOU SURE?"

It is possible for some programs to alter the computer's memory so that you will not be able to LOAD programs from the menu correctly until you reset the machine. We therefore suggest that you turn your computer off and then on before loading each program.

Copying the programs

The disk is not protected in any way so you can copy the programs onto your own disk should you wish. All of the programs can be loaded independently of the menu by following the instructions with the relevant article.

Disk Problems

Should you have problems loading any of the programs on the disk DO NOT return them to the editorial office. All faulty disks should be returned to:

Commodore Disk User Vol 1 no 3 Returns
Disk Copy Labs
20 Osyth Close
Brackmills Industrial Estate
Northampton
NN4 0DY

and a replacement disk will be sent to you.

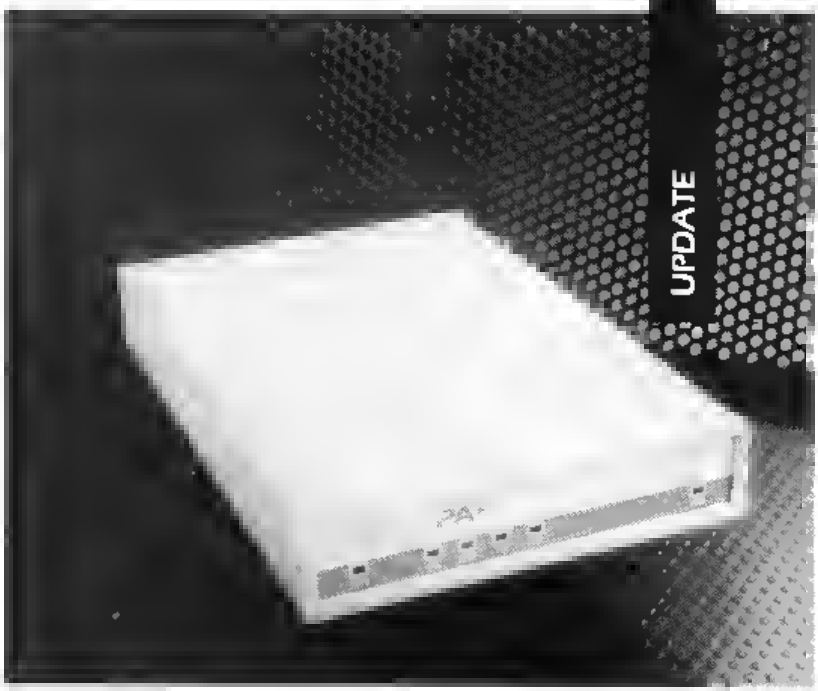
Zeppelin takes off

Recently formed Newcastle software house Zeppelin Software has just announced its first releases. The games will initially be released on cassette at £2.99, although it is likely that disk versions will follow.

Zybex is a smooth-scrolling game which requires you to find your way through a number of levels while acquiring new and more powerful weapons at each stage. Zeppelin says that there are hundreds of variations on the basic gameplay.

In Sabotage, you play a mercenary who has to get to each sector of a planet and destroy any alien attack craft which approach. At each sector you can collect a piece of a blueprint and a password which lets you go to the next level. At the end of the game all eight pieces of blueprint are assembled and a secret computer code revealed.

For more details contact Zeppelin on (091) 281 4401.



Modem Package from Telemap

Telemap, the company which operates Micronet, is launching its own modem package. The aim is to make its online services cheaper and more convenient to home and business micro users.

The package, known as ProPak, consists of the Telemap ProPak modem, plus machine-specific software, connecting cables and a year's subscription to Micronet, Prestel and Telecom Gold. The modem is Hayes-compatible and supports auto-dial and V21 and V23 protocols.

The cost of the whole package is £199 plus VAT. This comes out to be £218.95 since subscriptions are not VAT-rateable. The company says that this is a saving of £168 on normal retail prices.

Telemap's online service, Micronet is the largest information provider on Prestel with over 20,000 subscribers. For more information contact Telemap on 01-278 3143.

Budget printer from Brother

Brother has launched a new budget printer. The Brother 1209 costs only £265, yet has a print speed of 168 cps. It can be used with single sheets or continuous stationery and the company says that it is ideal for use in educational establishments as well as for business applications.

There are three resident fonts - Prestige, Gothic and Quadro and most printing functions can be performed via a control panel at the front of the machine.

The 1209 can also emulate the Epson EX/FX series of printers and the IBM Proprinter XL. This makes it compatible with all popular software. For more details contact Brother on 061-330 6531.

Mini Office is a Moneyspinner

Channel Four's financial programme Moneyspinner, has voted Database Software's Mini Office II to be Britain's top home business package. Runners-up were Supercalc 3, Money Manager Plus and Planit.

The entrants were judged on the basis of three criteria value for money; ability to do the job, and ease of use. Said Belinda Giles, Moneyspinner's producer, "We were looking for a package which would be most suitable for viewers attempting to produce accounts on home computers. Mini Office II did everything required of it, and it was truly easy to use."

Mini Office II is available for £19.95 from Database Software on (0625) 878888.

EA battles it out

Electronic Arts has announced the imminent release of two new World War II simulations, including the biggest naval battle of all time.

In EA's Leyte Gulf, the player assumes the role of commander of the US fleet charged with the invasion of the Philipines at Leyte Gulf. The player's fleet consists of a battleship, aircraft carrier and cargo troop ship.

The game comes in two parts, direction of the battle from the bridge and combat simulation scenes. On the bridge the commander can access four ship departments, navigation, communications, fleet status and weapons operations to determine overall strategy.

The battle sequences include ship-to-ship action, ship-to-air, which includes fending off Kamikazes, air-to-ship, the launch of air strikes from carriers, and ship-to-land, the actual invasion.

A simultaneous release is The Train: Escape to Normandy, based on a famous film starring Burt Lancaster. The Train requires you to commandeer a French train containing France's finest art treasures which have been looted by the Nazis, and manouevre it successfully through enemy lines to the Allies at Riviere in Normandy.

The player has the aid of Le Duc, an injured resistance fighter, and has to stoke the engine and set switches and set the correct boiler pressure. En route, many hazards are encountered such as enemy traps, ambushes and sabotage. 50mm cannons at front and rear of the train allow combat with enemy planes, artillery and gunboats. Le Duc gives the player advice, direction, inside information, enemy alerts, and relays messages from the Resistance.

Both Leyte Gulf and The Train are available from Electronic Arts at £14.95. For further information contact Electronic Arts on (0753) 49442.

▶ ▶ ▶ ▶ ▶ ▶ ▶

Elementary?

Infocom has released a new adventure based on the adventures of Sherlock Holmes Titled Sherlock The Riddle of the Crown Jewels, it actually gives poor old Watson, who is played by you in this case, a chance to solve a case for himself

In the game, the Crown Jewels have been stolen from the Tower just two days from the Queen's Jubilee To find the thief, Watson has to decode a trail of infuriating riddles Holmes will condescend to assist with advice, of course

On-screen hints are available, and the game comes complete with a tourist map, a newspaper and a key fob

Sherlock The Riddle of the Crown Jewels is part of Challenge Inc's Immortal Legend series It will be available at £19 99 for the C64 For for further information contact Activision/Infocom on 01-431 1101

Mastertronic extends its range

Budget software house Mastertronic has cut an important deal with Activision, publisher of Rescue on Fractalus and Ghostbusters The deal will allow Mastertronic to market all of Activision's older games at budget prices The prices for cassette-based games on the C64 will be £1 99 or £2 99

The deal covers future Activision titles, although not the important Infocom range of adventures The first batch of titles is now on release and includes Ghostbusters, Eidolon and Ball Blazer For more details contact Mastertronic on 01-377 6880

Protect your power

The Supa 7 is a new device launched by Warwick Products to protect all home and office equipment from electrical interference through the mains

The unit is rated at 7 amps and performs two functions preventing voltage spikes from getting through and filtering out radio frequency interference, which is a major cause of electronic malfunction

The Supa 7 is available for £14 95 plus VAT and 50p p&p from Warwick products on 01-538 2353

Hats off for Evesham

Evesham Micros is giving C64 users the chance to win an Amiga - and, perhaps, to see the company's director eat his hat The company is offering the prize to any entrant who can find a piece of commercially available software which is compatible with the Commodore 1541C disk drive, but not with the new version of the Evesham Excelerator

Evesham is anxious to ensure that the Excelerator is compatible with all C64 software The drive at, £159 45, is £40 cheaper than the Commodore drive There have however been fears that a number of pieces of turboload software would not load on the Evesham drive

Richard Austin Evesham's managing director (above), who has promised to eat his hat if the company has to give away more than two Amigas, said, "We have tried to make the Excelerator as efficient as possible, but we cannot test every computer game because there are too many of them. Who better than the public to attempt this task for us?"

Entrants have to write down on a postcard the name of a C64 program which can be obtained from a computer shop and which is compatible with the 1541C but not with the Excelerator The first entry received by March 1988 for each different program will win an Amiga Entries should be sent to Evesham Micros, c/o Solution Public Relations, Carlton House, 11-12 Marlborough Place, Brighton

A Star is Born

Star Micronics has launched a successor to its successful NL-10 dot-matrix printer. The new machine, the LC-10, has, says the company, a higher specification for a lower price.

The LC-10 is a multi-font printer with dual printing speeds of 120 cps in draft mode and 30 cps for NLQ output. There are six resident fonts and seven print pitches, all selectable from the front panel. Also standard is a push-feed tractor feed, and continous stationery, does not have to be removed to use the single-sheet feed facility.

The LC-10 is available at £229 ex. VAT from Star Micronics UK Ltd on 01-840 1800.

Reviews

Skate or die!

If you think that skateboarding is dead, then think again, as Electronic Arts challenge you to Skate or Die! Five events will check your skill and style as you take to the half pipe, the streets and an emptied pool for a joust with the town's top skaters.

Once the disk is loaded it's time to check in at Rodney's Skateshop to grab a board and your place in the action. By moving a "Skate and Die" cursor around the screen you can sign up for the tournament (along with two other humans), check out the high scores, pick up some tips from the man himself, select a board from the 15 available colours or practise some of the events.

When you leave the shop the screen is replaced by a scene of a street corner where you can try out your basic board skills before heading down one of the roads that lead to single events. The most challenging road leads to them all.

A hundred feet of wood and steel forms the U-shaped ramp, scene of the freestyle event. The crowd is excited as you stand on top of the ramp ready for your ten passes down the ramp and a chance to show off some well wicked moves. As you hurtle down the ramp pressing the button and moving the joystick you can amaze your fans and rack up the points with a succession of kickturns, footplants, rail slides, handplants, aerial and reverse Ollie air turns. You get just ten passes through the ramp to amass as many points as possible so you have to take risks and mix up the action since nobody is going to get that excited about ten kickturns! As you leap or turn in the air you have to be sure that you land properly or you'll get stoked which means to get separated from your board and land so heavily that it will knock your socks off as well as your arms, legs and head!

The high jump event couldn't be simpler. After all you only have to build up as much speed as possible then leap as high up a bar on the right of the ramp. You can even add a few extra inches by pressing the joystick button at the highest point of the jump. Unfortunately if you mistime any of these you'll be sent crashing to intensive care.

If you prefer your action out on the street then sign up for the downhill jam 'inter city back alley blazing!'

This event will not only test your skating skill through an alley that's an obstacle course of trash cans, fences, boxes, manholes and buildings but also your toughness as you punch and kick your opponent to pile on the points. If you've got time you can grab some boosts by kicking over trash cans, busting bottles and smashing flower pots. Whatever you do, make the most of it before you meet the reception committee in the shape of the police waiting at the end of the alley.

Out of the alley, and into the park - it's a quiet day without a brat, worker or park ranger in sight! Which means it's prime time for thrashing! The object of this event is to race through the park as fast as you can while taking on the opportunity to skate or die! Are you brave enough to skate through the pipe, leap over the jumps or even hurtle over the lake to get the top score?

Finally it's off to the pool, not for a swim but for the pool joust. Two skaters, an empty pool and one boffing stick are the ingredients required and if you haven't got an opponent you can take on Poseur Pete, Aggro Eddie or the Lester the local champ.

Each skater takes it in turn to have the boffing stick. They have five passes to boff their opponent who obviously must skate out of the

way to keep their lives intact for when they get the stick. You'll need a cool nerve and superb timing to stay on your board.

Skate or Die! is one of the best disk games you can buy - it simply oozes quality. The five events each pose a different challenge, from the acrobatics of the freestyle ramp, to the sheer guts of the downhill race and pool joust. These games are backed up with impressive graphics, from the colour of your skateboard, to the way your skater adjusts his shades before starting down the downhill run. Even the screen where you select the events is well thought out and gives you a chance to practise your skills before you Skate or Die! **TH**

AT A GLANCE

Name: Skate or Die!
Supplier: Electronic Arts, 11/49 Station Road, Langley, Berks, SL3 8YN
Tel: (0753) 49442
Price: £14.95
Graphics: Superb
Sound: Superb
Sound: Skate, rattle and roll!
Playability: great
Addictiveness: wicked

The Hunt for Red October

Based on the book by Tom Clancy, this submarine simulation casts you as Captain Marko Ramius, commander of Red October. Red October is the Russians' most advanced nuclear submarine and the first to carry the revolutionary caterpillar drive that allows it to move silently through the seas. On what appears to the crew as a regular mission to patrol the American coast you really plan to defect!

Having killed the political (KGB) officer you set course for your rendezvous with the Americans. It will be a dangerous journey however, as your former comrades are determined to stop you at any cost. Trawlers scan the seas for a contact and ships and submarines close the net on you. To add to your problems you will have to navigate through the treacherous trenches and peaks of the Reykjanes Ridge while avoiding the American mines.

Although the crew members are innocent to your real intentions your officers will get them to carry out your orders as long as you remain close to the course they're expecting.

You control the Red October through a series of icons arranged at the bottom of the screen, below a map area that can be toggled between either a sonar or contour map to guide you through the ridges and a 3D periscope view that is also used to lock onto torpedo targets. The icons are arranged in groups so you can quickly reach the one you want. For example selecting the engines icon produces a new set to select nuclear, diesel, caterpillar or propellor power.

Red October's speed, heading and depth are controlled by pointing and clicking the hammer and sickle on a separate display and then altered by clicking increase or decrease. This combination of controls work well and allow you to quickly alter your tactics whenever your sonar crew warn of a mine or an approaching vessel. This you must be ready to do at all times, especially as the Soviet network of trawlers, ships and submarines closes in.

Your tactics will depend on your nerve and your skill. Can you weave your way through the peaks so tightly it makes it impossible for the enemy to track you? Or will you run for the open sea at full speed and fight it out with anything that comes too near? You could even dive to the bottom and get behind your hunters although it will be almost impossible to reach the Americans. Whatever your tactics, you must remember not to fire on a single US ship as the Americans will immediately join the Russians in their seek and destroy mission.

If you are to have any chance in this game at all you will have to forget every other submarine game you've played before where you have hunted and killed enemy convoys. This time you're the one being hunted! To add to your troubles if through your actions the crew become suspicious they will revolt and you'll be scuppered.

At the end of the game the New York Telegraph will report on the outcome, however Russians destroy their own submarine or the most likely headline. An enjoyable but difficult game that adds a new twist to the submarine simulation. **TH**

AT A GLANCE

Name: The Hunt for Red October
Supplier: Grandslam, Victory House, Leicester Place, London, WC2 7NB
Tel: 01-439 0666
Price: £14.95
Graphics: Good
Sound: beep, bèep, bang!!
Playability: easy to play but hard to win
Addictiveness: very

Airborne Ranger

Armed to the teeth, and trained to kill the Airborne Ranger does the jobs that somehow must be done. Thanks to Microprose, you can sign up as one of the elite and brave. Twelve different missions are offered in arctic, temperate and desert conditions.

Once you have selected a mission and difficulty level (from difficult to impossible) you're dropped by parachute behind enemy lines but before you jump you can drop three bundles of supplies on your expected route as the plane flies over the battlezone. These supplies are important as they contain extra ammunition for your rifle, more grenades, rockets and medic kits to patch up your wounds. If you tried to carry them all at once it would slow you down too much making you a sitting duck for the enemy troops, machine guns and mini tanks.

Depending on your mission briefing you will have to destroy a plane or a fuel dump, photograph a new weapon or free prisoners, kidnap an enemy commander or cause a diversion by wiping out everything in your path. Whatever your target, it will be at the other end of a battlefield that's literally crawling with rifle-shooting enemy soldiers, minefields and gun emplacements. In some missions your job is made even tougher as you can only strike with your knife so as not to sound the alarm.

When you've studied the lay of the land you're ready for action. You can move through the terrain in three different ways. You can either hide from the enemy by crawling along the ground, or walk or run straight at them with guns blazing to catch them by surprise. Unfortunately, you can't spend the entire mission on your belly or you'll miss the rendezvous point and since you will tire too quickly if you run everywhere you must use all three speeds as and when they're required.

Airborne Ranger puts you in the thick of the action with impressive 3D graphics and challenging computer opponents. To reach your target you will have to avoid or deal with guards that actually patrol and machine gun nests that shoot on sight. In the meantime you are busy keeping out of minefields and finding a way through barbed wire and around walls, past proximity mines until you finally reach your target.

To survive, you will have to learn how to creep up and knife a guard, shoot it out when desperately outnumbered, silence a machine gun nest with a single grenade and create diversions with your rockets and time bombs that will send the guards running away from you or into a planned ambush.

With your mission completed you will receive a report, score and maybe even a promotion or medal, and finally you go on to the Congressional Medal of Honour.

Airborne Ranger is a game that you will have to practise to realise its full potential. Your first attempts will end in failure either as a result of enemy fire or stepping on a mine but soon you'll climb into the action and "feel" your way through. You'll then be hooked and driven on with a thirst for glory and promotion. Only the most experienced Rangers, however, should push the difficulty to its limit as the higher levels replace the rifle-firing guards with ones armed with rockets and grenades. This is the elite of the combat games. TH

AT A GLANCE

Name: Airborne Ranger
Supplier: Microprose, 2, Market Place, Tetbury, Glouc, GL8 8DA
Tel: 0666 54326
Price: £19.95
Graphics: action packed
Sound: bang, bang, boom!
Playability: climb into the action
Addictiveness: I'll get them next time!

Scruples

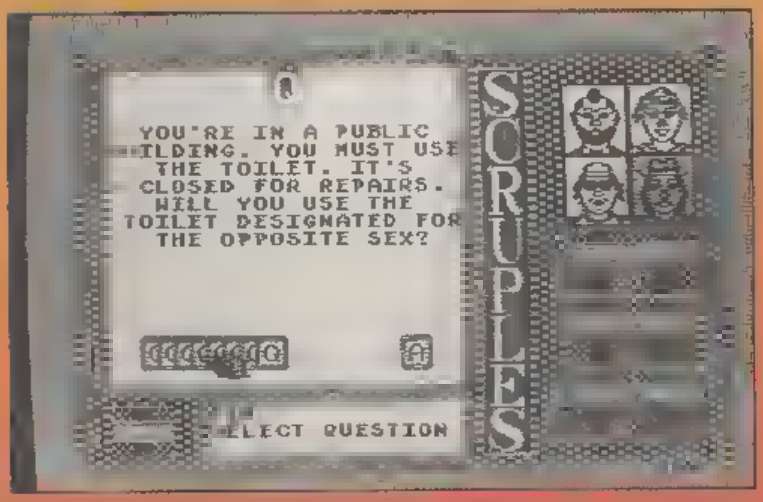
Billed as the game of moral dilemmas, A Question of Scruples has you answering questions such as would you keep an expensive pen that you found lying in the street or, whether or not you would have an affair if your regular partner were away for a long period of time.

Giving the right answer is not necessarily the thing to do though. It depends on how much you want to win! The trick is to give an answer that disagrees with the answer predicted by the questioner and then be able to support your decision if challenged. The other players then vote on whom they believe.

The game mechanics are fairly simple. Each player is dealt a number of 'dilemma' cards which have the problems on. The object is to get rid of these as quickly as possible. You are also dealt an answer card which must be kept secret from the other players. The key is to ask people questions that you think they will respond to in the same fashion as shown on your answer card.

The computer version of the game is for 3-10 players. These can all be human or a mixture of human and computer opponents. Each human must begin by defining their personality by setting levels for twelve different traits on a histogram. These traits include personal and professional integrity, greed and honesty. At the end of the game, you are given an updated histogram based on the answers that you gave.

The game features lots of variety. There are 64 different computer opponents ranging from gangsters to vicars. These characters have built in levels both of competitiveness and intelligence. This means that they will try to win and try to predict your answers from the personality factors that you gave. All this combines to give as much variety as you are ever likely to need.



The mechanics of Scruples are extremely well-implemented and I couldn't find anything much the matter with the gameplay. There is a major problem though and that is that I don't think that Scruples is the sort of game that works well on the computer.

The main force behind the game is that it depends so much on human interactions and the computer just has the effect of depersonalising everything. Making people look away while you examine your answer card is annoying and trying to examine someone's eyes to see whether or not they are lying is very difficult when you are all staring at the screen.

To conclude then, a worthy conversion but a game that is best suited to sitting round a coffee table rather than a VDU. **GRH**

AT A GLANCE
Title: A Question of Scruples
Supplier: Leisure Genius, 2-4 Vernon Yard, Portobello Rd, London W11 2DX **Tel:** 01-727 8070
Price: £14.95
Graphics: Dubious-looking characters give dubious answers
Sound: N/A
Addictiveness: You still need a good supply of human opponents
Playability: No real opportunity to cheat (oh, what a giveaway)

Eye

Every year, a new strategy game appears on the market claiming to be simpler to play than Snap but more difficult to master than Chess. Remember Kensington, September, Continuo or Mandala? Maybe not, but I suspect that you have all heard of chess. Doubtful publicity claims aside, this year's contender is Eye.

As with most strategy games of this nature, the object is to move your pieces round the board into a winning position before your opponent does likewise. Nothing original in that, you say. But it is the board that makes you sit up and take notice for that really is different. It is also very difficult to describe without seeing a picture of it in front of you.

The board consists of two rotatable spirals sitting on top of a coloured baseboard. The arrangement is such that there are always thirty-two coloured squares in view of eight different colours. The fun is that as part of your move, you can twist one of the spirals thus changing the colours of the squares below.

Each player has a different colour and the object is to get an appropriate number of your pieces on your own colour. Play starts by taking it in turns to position your counters on the board. The number of moves that you have depends on the highest number of counters you have on any one colour. A move can consist of moving a piece to an adjacent square or turning the board. Should you control a colour other than your own you also get the opportunity, before you start moving your pieces, to move any of your opponents' pieces that happen to be lying on the controlled colour.

The game is for two to four players although I felt that the two player version worked best. One extremely annoying feature is that you have to set up the playing conditions every time you play. A simple yes/no choice to change/keep the parameters should not have been too difficult to include. The computer version has several playing options available not possible in the board game but even so, I preferred the board version. There is something very satisfying about physically turning the spirals. Whether I will still be playing it next Christmas though, eye have my doubts. **GRH**

AT A GLANCE
Title: EYE
Supplier: Endurance games, Unit 1, Baird Rd, Enfield, Middx EN1 1SS
Tel: 01-804 8100
Price: £14.95
Graphics: Clear and colourful
Sound: N/A
Addictiveness: Initial interest then rapid decline
Playability: Setting up is a pain, the rest good

Combat School

Combat School has been a megahit on cassette and it's easy to see why It's tough, varied and exciting but in some ways it's too difficult.

As a trainee marine you might expect that the training course will be challenging and this must be the toughest challenge around Obstacle courses, several shooting ranges, cross-country running and canoeing, arm wrestling and unarmed combat. Each

challenge has a time limit or a qualifying score which is often so tightly calculated that nothing but a near perfect performance will allow you to pass onto the next stage.

Success must be attained at every stage because failure at any one point takes you back to the start. A near miss, such as one shot short of a qualifying score on the firing range, results in a second chance By performing a number of chin-ups in a given time you can avoid being drummed out but each time you have to resort to this challenge it gets harder and harder

Once you've competed successfully in all of the events, you can go on to the second part of the game where you must complete a secret mission to rescue a hostage in the American Embassy The hostage is guarded by a tough team of terrorists and my first visit lasted about three seconds This is annoying because failure means going back through the school course again and a code to allow you to attempt the mission after disqualifying once would have appealed to me more

Despite this, Combat School is excellent - probably the best thing to come out of Ocean's liaison with Konami It's certainly one of the best games that Ocean has produced in its long and eventful life **NJD**

AT A GLANCE

Name: Combat School
Supplier: Ocean, 6 Central Street, Manchester M2 5NS
Tel: 061-832 6633
Price: £12.95
Graphics: A1
Sound: Strained and fuzzy at times
Playability: Very, very mean
Addictiveness: Can't put it down

Out Of This World

While playing around with trying to improve on a maximum speed of 0.9 times the speed of light, Captain Chuck Harrison has blasted himself into an alternative universe. So goes the scenario for this game on the Reaktor label from Ariolasoft.

Waves of aliens move across the screen at different speed and from different directions. Dodging and weaving your craft through the waves, your aim as Captain Chuck is to kill as many aliens and collect the counters that they drop to improve the weaponry of the RP2-16 craft that he's flying in

Not all of the alien ships drop counters but those that do leave these tokens bouncing wildly around the screen. Trying to collect them is a nightmare as more waves of aliens appear zig-zagging or gliding across the scrolling backdrop

When enough tokens have been collected, they are automatically cashed in for a weapon selected from a range of seven alternatives.

Out Of This World is just another version on the Nemesis theme but lacks the megaships which made the real challenge of the arcade original. This offering has the look and feel of a budget label game and if it had cost £5 99 the review would have been different.

On the plus side, it does play quite well but can't overcome my feeling that it lacks the challenge that I would expect of a £12 99 game

Ariolasoft has come down in the world since the heady days of Broderbund and Electronic Arts label games and this is not the type of game which will bring them back again. Nice graphics though **NJD**

AT A GLANCE

Name: Out Of This World
Supplier: Anolasoft, 9 Disraeli Road, Putney, London SW15 2DR
Tel: 01-785 4285
Price: £12.99
Graphics: Hunky
Sound: Ample but not inspiring
Playability: Fast and furious
Addictiveness: Fairly average

Gary Lineker's Superstar Soccer

Have you got what it takes as a player to score vital goals in important matches? Have you what it takes to manage and build a team that will rise from the foot of the fourth division to League and Cup glory? You will need both to succeed in this football simulation named after England's top striker.

Your job as manager of your chosen team is to pick the team, build the squad, make decisive moves into the software market and decide the tactics during the games. As a player you can be either the goal scoring centre forward or the last line of defence, the goal keeper. You will also have four other computer-controlled team mates that are directed by the management tactics.

Each player has an age and skill factor that determines not only how well they play but also how quickly they move. As the game begins you have a squad of teenagers that you must build into a championship team and take to as much glory as you can cram into nine years. To do this you will have to spend your trading points wisely - these are allocated at the beginning of each season and can be used to send your whole squad to training camp (slight improvement in most players), train a youngster (you get a 17 year old with a useful skill level) or to trade with another team by offering a player and some trade points. If your offer is accepted you will have a new player, if it's refused you can try elsewhere but you will have lost some points through arbitration.

The football matches themselves play like a miniature version of International Soccer and can be remarkably tight affairs, but unfortunately you'll have plenty of time to decide your tactics and celebrate your victory or wallow in defeat as the game takes ages to load and save game data. If you can cope with that you'll have to endure the annoying Americanisms that pollute the game, as you realise that Superstar Soccer has more to do with Joe Montana than Gary Lineker. For example, the play-offs replace the cup, you'll play overtime instead of extra time and be awarded free kicks not for obstruction or fouls but for "delay of game" and "interference"!

If you can cope with all that, and I could, you're left with a highly enjoyable game that will keep you playing well into the small hours.

TH

AT A GLANCE

Name: Gary Lineker's SuperStar Soccer
Supplier: Gremlin Graphics, Alpha House, 10 Carver Street, Sheffield, S1 4FS. **Tel:** (0792) 753423
Price: £14.95
Graphics: matchstick men
Sound: cheers and whistles
Playability: good but takes an age to load

Flying Shark

At last the true arcade version of Taito's Flying Shark has been translated to the C64 and it is superb. Firebird's programmers Dominic Robinson and John Cumming have produced a brilliant game which shoots 1942 and the like down in flames.

I don't believe the exaggerated boasts scrawled on disk inserts but for once here is a game which lives up to the promise of 'the definitive conversion'. Superb graphics, a wide range of enemy planes, battleships and tanks in all shapes and sizes means that flying the prop-driven fighter plane across the detailed, scrolling landscape never gets boring.

As far as rules are concerned the set up is simple: dodge the bullets, blast everything that moves and everything that doesn't!

As you blast your way through Hell, the occasional superweapon or a much-needed extra life bonus can be grabbed. You'd be best advised to grab whatever's going because after an eternity of sweaty survival tactics you'll need all the firepower you can muster to wipe out the megaweapons at the end of each level.

Death is not as fearful as with most games, painful though it may be. Each level is split up into stages and an unwise decision means reincarnation at the beginning of the current stage rather than at the start of the game.

Flying Shark receives my highest recommendation as an addictive and accurate arcade conversion. Well done, Firebird! **NJD**

AT A GLANCE

Name: Flying Shark
Supplier: Firebird Software, First Floor, 64-67 New Oxford Street, London WC1A 1PS
Tel: 01-379 6755
Price: £14.95
Graphics: Cute and colourful
Sound: Won't win awards but won't offend
Playability: Addictive to the extreme

ON THE DISK

Super-tact

A deceptively simple game of strategy and tactics, this 'll have you biting your fingernails in frustration in no time flat

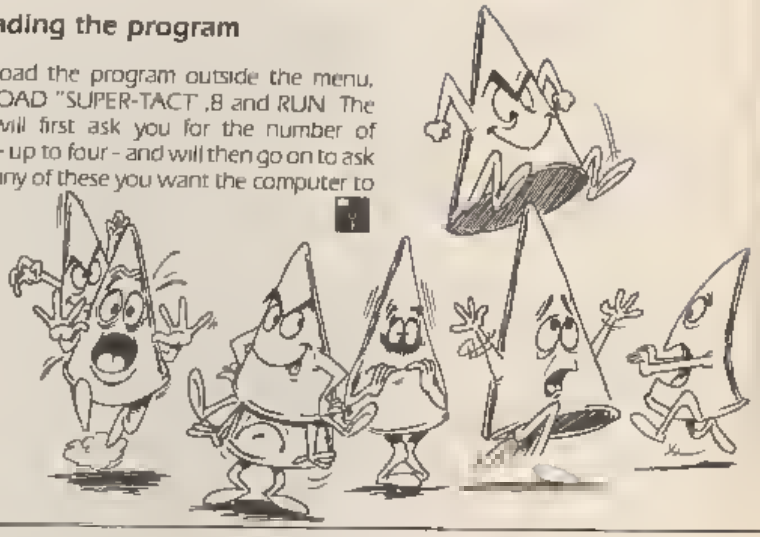
By R Kyme-Wright

Super-tact is a game for up to four players You can also specify whether or not your opponents will be computer or human The game requires you to move the counters of your own colour around a 'race track' to a home area You have four counters and so do each of your opponents The number of squares you can move is determined by a dice throw, but before you can move anything at all, you have to throw a six (after his you will get a second throw) The dice throws for you and your opponents are arranged by the computer, and appear automatically displayed at the centre of the screen

Simple but the catch is that if your counter is landed on by another of yours, or by one of your opponents, then it automatically returns to its start position and has to go all over again Of course, you can mess up the opposition's prospects in the same way The game operates on the wimp system You are equipped with a pointer, and you have to use the joystick to point it to the piece you want to move Beware, though, if you point it to the wrong place, you forfeit the move

Loading the program

To load the program outside the menu, enter LOAD "SUPER-TACT",8 and RUN The game will first ask you for the number of players - up to four - and will then go on to ask how many of these you want the computer to play



Chaos in space

Can you survive wave after wave of vicious alien attacks? Hone up the reflexes with this month's blast-em-away game

By Paul Williams

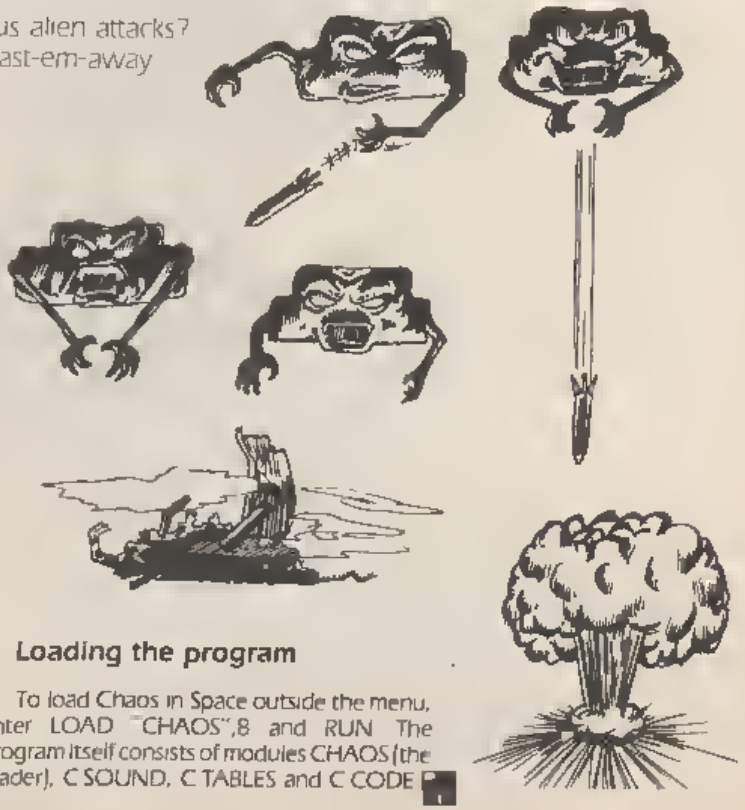
Chaos in Space is a shoot-em-up that may look a little familiar at first sight, but beware, it has a hidden sting in the tail It may appear to be classic Space Invaders, and functions very well in this role, but waiting for you in there is a little touch of Galaxian

The game is played with a joystick in Port 2 Pressing the Fire button will start the game, and the Stop key will pause it at any point Use the joystick to move your laser base right to left, and blast away at the aliens with the fire button Make your shots count - you only have three lives

Points are counted for aliens as follows

500 pts: Green Swoopers
700 pts Blue Swoopers
900 pts Red Swoopers
200 pts. Edge of mothership
500 pts: Centre of mothership

You only get 50 pts for an alien if it isn't swooping



Loading the program

To load Chaos in Space outside the menu, enter LOAD "CHAOS",8 and RUN The program itself consists of modules CHAOS (the loader), C SOUND, C TABLES and C CODE

Software Over the Rainbow

The weird multicolour system on the C64 can be difficult to understand. Here is the ultimate key to a colourful future

By Norman Doyle

Multicolour can be a difficult mode to understand and can also be difficult to apply. The problem is one of packing information into the screen and results in a loss in horizontal resolution or, in plain terms, a reduction in the number of pixels across the screen. Why this problem occurs can only be explained by looking at the way in which the computer applies colour to the screen.

Whether programming in Basic or machine code, the user soon becomes aware that there are two screen maps in their Commodore: one contains the value of the displayed characters and the other holds the value of their colours. The screen character memory starts at $0400 (1024 in decimal) and runs through 1000 locations to $07E7 (2023). The corresponding colour map starts at $D800 (55296) and ends at $DBE7 (56295). Unlike character memory which can be switched around throughout the memory and doesn't have to remain at $0400, colour memory always occupies the same locations and doesn't use its allocated space very economically. There are 16 colours available which means that only half a byte (a nybble) is needed to store the value of each character's colour.

This can be tested by POKEing a character and a colour to their respective screen maps.

```
POKE 1024,1:POKE 44296,7
```

This produces a yellow letter A on the screen. Change the seven to 39($27), 135($87), 247($F7), each time the result is the same yellow character. Notice that the hexadecimal value is a better guide to the colour which appears. If you peek into the location after you've poked the number the decimal value probably won't resemble the number you poked but converting it to hex you will always find that the low nybble will be correct for the colour displayed. This can be proved by using the following:

```
PRINT PEEK(55296)PEEK(55296)AND15
```

Each time this line is used the first value may differ from the value that was poked in but the second value will always be the current colour of the character in the top left corner. In other words, we are only interested in the low nybble of the colour map memory.

Colour Considerations

In multicolour mode the memory map locations represent only one of the character's colours but in a very peculiar way. To examine this power up the computer and type in this line:

```
PRINT"[CLR][[[[["
```

The screen clears and five brackets are printed.

```
POKE53270,PEEK(53270)OR16
```

Now the computer goes into multicolour mode, but the brackets and the cursor disappear. To get the cursor back hold down the CTRL key and press the number 2 key and a white cursor will appear. As you type the following, notice that the letters now appear in normal mode and not multicolour:

```
FOR A=55296 TO 55300:POKE A,9:NEXT
```

We have POKEd the value for brown (9) into the colour memory map but the result is that the brackets appear as white characters. How is this phenomenon explained?

Firstly, the colour we are controlling is Multicolour 3 (m). In multicolour mode the value in the map can have one of two states - above or below a value of seven. If the value is seven or less the colour follows the normal rule according to the value used and the character is displayed in normal mode. If the number is eight or more the colour of Multicolour 3 becomes the value of the colour nybble minus eight. Nine was the value which was poked in and nine minus eight gives one - white.

It is obvious that this system limits the possible number of colours to the eight listed on the keyboard, an imposed limitation which is rigid and not avoidable by clever programming.

Colour Selection

The eight colour limitation does not apply to the other two colours which multicolour mode can access, the value of these is stored in locations $D022 (53282) for Multicolour 1 and $D023 (53283) for Multicolour 2.

Any of the 16 colours can be used but the limitation is that these colours are the same for all the characters on the screen. The only exception to this rule is when advanced programmers use split-screen techniques when these locations can be changed back and forth with each interrupt routine.

This limitation means that the most colourful non-interrupt screen displays can only be achieved if the programmer uses Multicolour 3 for the dominant character colour and Multicolours 2 and 3 for the less noticeable colours.

To get back to the original problem of why the resolution is halved in multicolour mode, the answer to the next question supplies the solution: how does the computer know which colour to use?

In normal mode the colour can only have the value held in the memory map. Whether this colour is displayed depends on the 'state' of the pixel.

In multicolour mode the colour can be any of the three multicolours (pixel switched on) or the background colour (pixel off). The four states are selected by grouping pixels into pairs. Looking at this from the binary angle, each of the pixels can be either off or on. In our zero and one display the possible permutations of the pixel pairs are 00, 01, 10 or 11. Although there are still eight rows of eight pixels forming each character block, the fact that two bits are needed to describe the programmer's colour requirements to the computer's operating system (the VIC chip) means that neighbouring pixels must have the same colour. In other words, the perceived resolution is eight rows of four double-width pixels.

Colour selection broadly follows the same rules in multicolour as in normal mode - if either pixel within a pair is switched on, both pixels will be displayed in the same colour. If both pixels have a zero value, both are off and the background colour shows through.

This means that the three modes which are of most interest are the ones with at least one pixel in the 'on' mode: 01, 10, 11. The 11 mode is analogous to the pixel being switched on in normal mode, that is the colour is derived by the value held in that character position on the colour memory map. A binary value of 11 converts to 3 in decimal, hence why I've chosen to call this Multicolour 3.

In a similar way 01 and 10 translate to decimal as 1 and 2 and so both pixels will be displayed as either Multicolour 1 or Multicolour 2 respectively.

The Reality

In the example using the square brackets, the character information was not devised with multicolour mode in mind so the on or off states of the pixels is still the same when multicolour mode is switched on. How this affects each square bracket character can be seen in Diagram 1 where the pixel pairs have been drawn in.

As you can see the result is that every pixel has a 11 configuration. The normal mode colour was light blue which has a value of 14. In multicolour this value still determines the pixel colour of a 11 pair but, because of the seven colour rule, the perceived colour is eight less than this value which is 6 or blue. The background is also blue so the pixel colour is the same as the background and the character is 'invisible'. This can be proved by poking 14 back into the top five screen positions, the bracket characters will disappear but by poking the background colour to white they can be made to appear in their true colours.

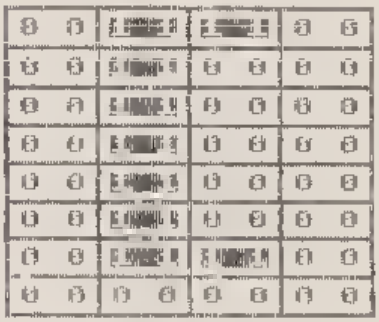


Diagram 1

Character Evaluation

The pixels are paired up but calculating byte values for the character rows still follows the same regulations as for user defined graphics (UDGs) in normal mode. Each pixel position has a value corresponding to 128, 64, 32, 16, 8, 4, 2, 1 (left to right). If the first two left-hand pixels are turned on and the others are off, the poke value would be 128 64 or 192.

It follows that 192 is also the value for selecting Multicolour 3 for this pixel pair. If 128 is POKEd both the pixels will be in Multicolour 2 and 64 will result in a Multicolour 1 pair.

Quick Change

Whether in multicolour or normal mode it's often desirable to colour a full screen or a block with one particular colour. Contained on the disk is a program called MULTIFILL which will fill part or all of the memory map with a specified colour. To use it poke location $02(2) with the colour value (remembering to add eight if Multicolour 3 is used). The width and height of the column is poked to $FA and $F8 and the start location is poked to $FC and $FD in low byte high byte order.

The routine is then called by JSR $033C from machine code or SYS828 in Basic.

The routine will only fill rectangular blocks but experimentation using several calls to this routine using different parameters will show how oddly shaped areas can be filled. I would recommend practising in normal mode before resorting to multicolour mode. The Basic example program MULTIFILL DEMO shows how these odd shapes can be defined.

C-Zap

If you feel the need for speed, convert your Basic programs to machine code with this powerful compiler

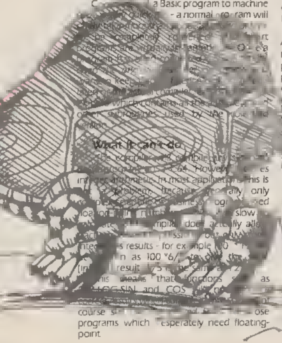
By Paul Williams

The C-Zap Basic compiler is a 12k chunk of machine code which takes any standard Commodore 64 Basic program and converts it entirely into machine code This dramatically speeds up programs, making, for example, Basic games run at arcade game speed A Basic program which normally takes about 3 minutes to run will, once compiled, run in about 10 seconds using this program

The reason for this is that normal interpreted Basic (as found in the C64 Rom) has to decode each line of your program each time that line executes For each GOTO it has to find the relevant line to jump to, and for each variable (e g A×B *C) it has to find the position in memory where that variable's value is stored, so Basic is very slow

A compiler does everything at compile time The whole program is converted into machine code which can be executed without needing the original Basic interpreter You get a program which can be directly run by the processor, which means it runs fast.

C[illegible] a Basic program to machine [illegible] - a normal program will [illegible] be completely [illegible] programs and [illegible] O[illegible] a [illegible] which contains all the [illegible] other subroutines used by the [illegible] version.



What it can't do

The compiler will compile any [illegible] 64. However [illegible] integer arithmetic. In most applications this is [illegible] problem, because [illegible] only [illegible] floating-point [illegible] slow [illegible] compiler does actually allow [illegible] integer results - for example [illegible] as 100 *6/[illegible] result [illegible] This means that functions [illegible] as [illegible] LOG SIN and COS [illegible] course [illegible] those programs which desperately need floating-point

One command which has to be modified is RND(1) See Appendix 1 for use of RND with the compiler All other standard BASIC commands work in the normal way, apart from the minor limitations listed below.

1 All numbers are stored as integers in the range -8323072 to +8388607

2 Strings can be any length up to 79 characters (this can be varied - see Appendix 2).

3 Arrays can be one or two dimensions, e g. DIM A(45) or DIM BD$(5,6). If an array is used without a DIM statement, it is assumed that the array is one-dimensional with elements 0 to 10 Generally, all arrays should be declared with a DIM statement before they are used For the compiler this means that they are declared physically before they are used, not in some subroutine at the end of the program which is called by a GOSUB This is because the compiler scans through the program in line order while compiling, and needs to find a DIM before a reference to an element in array

4 Because pure machine code is generated, the stop key is inoperative once a compiled program is running To halt such a program, press RUN/STOP and RESTORE together You can actually test for the stop key in a compiled program, though, with the following

GET K$:IF K$="[Stop key]"THEN END

The [Stop Key] will appear as a reverse-screen C when you type this line in

[illegible] compiler [illegible] a bug inherent in many [illegible] modore [illegible]'s: POKEing a character to the screen now does not require a similar POKE to the relevant colour memory byte for the character to appear, provided that a PRINT"[Clear-screen]" has been done since the last screen-scroll This feature works for [illegible] and Basic, provided that the compiler [illegible] compiler core] is resident and has been initialised (by running C-ZAP or the compiler, or some compiled code - see later)

Some Basic programs disable the operating [illegible] the system I/O page in order to access the character set, for example They usually re-enable the Rom or I/O with a POKE 1,58 which enables all the Rom Since

the compiler switches out the 8k Basic Rom to store the variables there, such a command would be disastrous to compiled code. For compilation, replace a POKE 1,58 with POKE 1, PEEK(1)OR6. (This is also fine for normal interpreted Basic, as it doesn't affect the status of the Basic Rom, just the Kernal Rom and I/O).

Features of the compiler

The compiler allows raw machine code to be included as part of a Basic program. This is a feature included for the programmer who knows about machine code who needs certain procedures to run at their fastest possible speed.

Machine code is included in a compilable program using the REM statement followed by a #, as in the following example:

```
100 REM#$A9,$00,$8D,$00,$04
```

The $-signs signify hexadecimal numbers. Upon execution of the compiled program containing this line, when line 100 is reached the processor will execute those bytes as machine code (in this case LDA#$00:STA $0400).

The input does not have to be in hexadecimal: decimal numbers are entered as themselves (e.g. 169,0,) and two-byte decimal numbers can be entered as D(number), as in D65534 which would result in 254,255 being stored (i.e. low byte followed by high byte).

See the example program - REM MC on the disk for a demonstration of this feature.

If you run a program containing REM# lines in normal interpreted BASIC, the machine code lines will be totally ignored as normal Basic skips over any REM statements. The REM# feature is only for use with compiled programs.

However, when writing programs from scratch for use with the compiler, the REM# feature is a handy way of including tables of numbers without using DATA statements (which take up 3 bytes per entry regardless of size).

```
4000 REM# 42,54,45,67,42
```

will place these five numbers in memory where line 4000 would normally be compiled. In order to access such a numeric table, the & sign is used. In an arithmetic expression, &(line number) returns the compiled memory address of the specified line number, so a line

```
10 FOR I = 0 TO 4 : PRINT PEEK(&4000+I), : NEXT
```

would result in 42 54 45 67 42 being printed to the screen when the compiled program is run. Again, this won't work in normal interpreted Basic - these are extensions to Basic solely for compiler use.

APPENDIX 1 - Use of RND with the compiler

One command which has to be modified is RND(1) - this normally gives a random number in the range 0 to 0.999999999 so this would be no good to the compiler. However, most of the time RND is used in the following way:

A=INT(RND(1)*X) or B=INT(X*RND(1))+1

To compile such a statement, just replace it by

A=RND(X) or B=RND(X)+1

i.e. the compiled version of RND(X) gives a random integer in the range 0 to X-1.

However, if you try to run this version in normal Basic, the function will revert to returning a value between 0 and 1. If you wish your program to run exactly the same interpreted OR compiled, use the following:

At the beginning of the program type

```
10DEF FNR(X)=INT(PEEK(56324)*X/256)
```

and replace every occurrence of INT(RND(1)*X) or INT(X*RND(1)) with FNR(X).

e.g. replace O=INT(RND(1)*100)+1 by O=FNR(100)+1

APPENDIX 2 - Changing the maximum string length

By default, the maximum string length permissible in compiled programs is 79 characters. This is usually no problem, as strings are rarely longer than that length (which is 2 screen lines).

However, for particularly large small-string arrays, it is sometimes necessary to define each array element to be smaller than 79 characters in order to fit all the elements of the array into memory, or in other cases a few strings might need to be longer than 79 characters. In order to change the maximum string length, type

```
POKE 49152,x
```

where x is one more than the maximum string length you want, just before compiling your program with SYS32768 or SYS32768,D.

Do not forget to either reload the compiler or POKE the string length back to POKE 49152,80 before compiling another program

Beware one pitfall - some very large string arrays cannot be squeezed into memory with the compiler so they write over the actual compiler code. This means that when the compiled version is run it will wipe out the compiler, causing no problem until you want to compile again, in which case you must reload the compiler with LOAD"C1",8,1 before typing another SYS32768 or SYS32768,D

APPENDIX 3 - Memory usage by the compiler

To indicate roughly what is going on when the compiler is active, there follows a list of memory usage

MEMORY AREA (Hexadecimal)	USAGE
$0000-$03FF	System workspace
$0400-$07FF	Screen memory
$0800-$1FFF	Basic program (can extend further if compiling to disk)
$2000-$xxxx	Compiled code when compiling into memory
$8000-$9FFF	Run-time string storage space
$B000-$BAFF	Run-time variable and array storage
$BB00-$BFFF	DATA storage
$A000-$BFFF	Standard Basic interpreter - switched out by compiler
$C000-$CFFF	Compiler run-time core and Dos Support system
$D000-$FFFF	Same as normal I/O and KERNAL ROM

Loading and running

To load the compiler type LOAD "C-ZAP",8 and RUN then either load or type in your Basic program To compile it, enter SYS32768

The compiler will get going, then either give you a table of memory allocation (ignore - it's just for interest) to indicate successful compilation, or report a compilation error

If the compiler has been successful - and generally if a program works properly in normal interpreted Basic the compiler will compile it successfully - you can now type SYS8192
to run the compiled version

As long as your Basic program is smaller than 7k, it will still be resident in the machine, so you can modify it if you like and re-compile If, however, your program is bigger than 7k you should not compile it directly into memory, but to disk by typing
SYS32768,D instead of just SYS32768

Then, to run the compiled version type
LOAD "PROG2",8,1
LOAD "PROG1"
RUN

This process can be used to save a finished compiled program to disk, and then each time you want to run it you don't need the compiler or your original program, so you can type (after LOAD"C2,8,1 (loads Compiler run-time core)
LOAD"PROG2",8,1 (loads any DATA used by your program)
LOAD"PROG1",8 (loads your compiled program)
RUN

If you like, you can rename PROG1 and PROG2 using @R0:NEWNAME=OLDNAME (the disk rename command) to something relevant to your program

So basically, use SYS32768 (if memory permits - and it usually will with most average-size Basic programs) to temporarily compile a program (and then run with SYS8192), and SYS32768,D to compile the disk, i e to create a permanent copy of the compiled code

None of the demonstrations with the Compiler are too large to be compiled directly into memory with SYS32768 - so by LISTing these you can get a feel of how big programs can be before restrictions are imposed

If your program has no DATA statements (signified by the compiler producing a line DATA STORAGE $BB00-$BB00 at compile time within the block of memory-allocation statistics printed) PROG2 is an empty file and it is not necessary to load it for future runs of the program.

The demonstration programs

A number of demonstration programs have been included on the compiler disk All of these are in their Basic form, so you can see for yourself the complexity of program that the compiler allows Each is loaded using / filename (or LOAD filename ,8) and can then be compiled using SYS32768 (assuming the compiler package C8OOT has already been loaded) - and executed with SYS8192

The demonstration programs present are

MAZE DEMO	A 3-dimensional view of a maze - the computer finds its own way through
MAZE	Same as above but you find your way through
HYPERTUNNEL	A pattern generator
ELEPHANT HUNT	A game which becomes arcade-game speed when compiled
WELCOME	Another pattern generator, this time using a programmable character set
BALLSA	A simple demonstration of bouncing balls
REM MC	A demonstration of the REM feature of the compiler for including machine-code in Basic programs
UNDERLINE	A useful program which converts the reverse-screen character set into underlined characters
DICE GRAPH	A simple statistical program plotting in 3-D the results of shaking two dice repeatedly
DIVIDE	If you MUST use floating-point this simple program divides one number by another to give an exact result to as many decimal places needed

After loading one of these programs, to see what a difference the compiler makes RUN the program in normal interpreted Basic, then compile it and run the compiled version

Basic+

C64 Basic is, well, a little spartan. Soup it up with this Basic toolkit

By Graeme Gayler

Basic+ adds a lot of useful commands to Commodore Basic - it certainly needs something adding! The goodies you will now have at your fingertips are described below. All commands will operate in the DIRECT MODE only.

Direct commands

@NUMBER n1,n2 This will renumber a Basic program from start to finish with first line numbered n1 and line spacing set at n2. This only numbers line numbers and not GOTO's and GOSUB's. If n1 or n2 or both are omitted default values of 10 are set.

@RENUMBER n1,n2 This will renumber a Basic program from start to finish with the first line numbered n1 and line spacing set at n2. This command will also renumber GOTO's, GOSUB's, GO's, and THEN's. If n1 or n2 or both are omitted default values of 10 are set. If the program to be renumbered has many GOTO's or GOSUB's, time must be allowed for the program to perform the renumbering as it might involve considerable amounts of memory moving to allow new line numbers to be inserted into the Basic text.

NOTE: There are two extra messages that can be flagged using the RENUMBER command -
1 'LINE SPACING TOO HIGH ERROR' which will occur if either n1 or n2 selected will create a line number greater than 63999 (maximum line number in Commodore Basic). This error is also active in the @NUMBER command.
2 'UN'DEF STATEMENT ERROR IN LINE * * * * *' Prior to renumbering the GOTO's the programme checks that all GOTO's direct to legal line numbers, if one does not the error will be flagged.
Both of these errors are nondestructive in that the checks are carried out prior to any alteration of the basic text.

@DELETE n1-n2 This will delete line numbers between, and including n1,n2. Syntax is exactly the same as for the normal LIST command, i.e. DELETE n1,DELETE n1-,DELETE-n2.

@LIST n1-n2 This command performs a Basic list in exactly the same way as the normal LIST command except that it LISTs to a printer only, if connected. Syntax the same as for DELETE.

@BIN n1 This will display the binary number n1 in decimal, decimal high/low, octal and hexadecimal format (n1 range 0 to 1111111111111111)

@DEC n1 This will display the decimal number n1 in decimal high/low binary, octal and hexadecimal format (n1 range 0 to 65335)

@OCT n1 This will display the octal number n1 in decimal, decimal high/low, binary and hexadecimal format (n1 range 0 to 177777)

@HEX n1 This will display the hexadecimal number n1 in decimal, decimal high/low, octal and binary format. (n1 range 0 to FFFF)

@PEEKn1 This command will peek at the location n1 in the same was as the normal PEEK command, but will display the address in decimal and hexadecimal, the number stored in decimal and hexadecimal. It will also display the condition of all the bits of the address.
Example of output from the @PEEK command
ADDRESS 49152 $c000
STORED 204 $CC
BIT 7 6 5 4 3 2 1 0

@OLD This command will restore a Basic program which has been lost with the NEW command.

@$ This command will display the disk directory on the screen without corrupting any programme resident in memory.

@"FILE NAME" This command will load into memory, from disk, the program FILE NAME. If the file name is left blank ie @" or @" " the routine jumps to the @$ routine and displays the directory.

Be careful

When BASIC+ is activated it changes the KERNAL LOAD ROUTINE VECTOR ($0330,$0331) to point to a new routine. This routine first checks the end address for a load from TAPE or DISK to see that it does not try to overwrite the BASIC+ program. If it does a '?OVERWRITE WEDGE ERROR' is flagged and all input/output operations are stopped. This protection will not work when the program that is being loaded alters the IRQ VECTOR STORE ($029F,$02A0) during tape input operations ie TURBO LOADERS.

@REVERT This will restore the computer to normal Basic and revert the KERNAL LOAD VECTOR to power up condition. BASIC+ can be re-activated with SYS 51200.

Loading the program

Basic+ can be loaded outside the menu by typing LOAD"BASIC+",8 and RUN. It can be restarted at any time by SYS 51200.

Tape archive

Keep your disk info safe - on tape - with this useful little program

By Les Allen

The floppy disk is a fast and convenient method for storage of computer software but is somewhat prone to accidental damage

By comparison tape is slow and tedious but robust in its construction and ideally suitable for producing a backup of a diskette

This routine provides the facility to download a complete disk of 664 blocks, in track/sector format, to tape in approximately 7 minutes. The routine does not transfer individual program files but creates a 'virtual disk' by copying each track and sector of the complete disk to tape

F1 DISK TO TAPE TRANSFER

Screen prompts are provided for insertion of the disk and destination tape, a unique tape header is provided that identifies the filename of the backup tape which can obviously be different to that given to the name of the disk. Press record and play to enable the process to be automatically completed without further operator involvement. Press RESTORE to return to the main menu

F3 TAPE TO DISK TRANSFER

The backup is not capable of being used on its own, and must be converted back to disk prior to use. Simply insert a previously formatted disk into the drive, the backup tape into the cassette, and follow the screen prompts. Press space when the tape header is found and the transfer process is completed without further operator intervention, in approximately 7 minutes. Press RESTORE to return to the main menu

This routine will not transfer disks that contain protection routines.

LOADING THE PROGRAM

To load Archive outside the menu, type LOAD "ARCHIVE", 8 and RUN

Link and Crunch

Tighten up your projects by combining the code and streamlining the storage

By Tony Crowther

Once a machine code program has been written it often occupies several different areas of memory and doesn't necessarily use the space as efficiently as it could. The Linker and Cruncher programs presented on this disk have been devised to help you correct this and save the need for Basic loaders.

Use of memory can be limited when writing a program because of the need to use relocated screen positions, user-defined character sets and other such reasons. When saving the final programs to disk you are left with two alternatives, either the whole area of memory which the units occupy can be saved or a Basic loader program can be generated to pull each section from disk. Both of these methods use up disk space which could be better used for other purposes.

The Linker allows you to store all the of a program sequentially in memory combined with a special routine which will pull the sections apart and relocate them to the correct areas of memory when your Linked program is loaded and run.

To create a master program you LOAD and RUN the Linker. This will display the input screen which uses special non-alphabetic symbols to issue commands. Anything else which is typed in on the screen will be treated as a program to be searched for and loaded into the Linker sequence.

Disk commands can be issued if just the part enclosed in quotes in the normal Basic commands are typed in, preceded by the '@' symbol (situated beside the asterisk on the keyboard) and stripped of their quotes. For example, @S0:MENU would erase the program called MENU from the disk. On its own, the '@' will print the contents of the drive's error channel on the screen. A directory can be displayed simply by entering the dollar symbol followed by a RETURN. This prints directly to the screen and does not affect the memory contents at all so it can be used at any time during the linking process.

If you're fortunate enough to have two disk drives, you can set them up as devices 8 and 9 and use the left pointing arrow at the top left of the keyboard to toggle between the two units.

If you make a mistake when loading the routines for linking, there is no other option but to clear the memory using the upward pointing arrow symbol, which is located to the right of the asterisk on the keyboard.

The final two commands relate to the linking process itself.

Program sections can be loaded by typing in the name of each section and pressing RETURN. No special load command need to be used and the program will automatically append each new program onto the end of the last one loaded into memory. If at any point you need to check what programs are currently stored the contents can be displayed by entering an asterisk (*). This gives the first four letters of each program name, their start and end addresses in the Linker's memory, their actual load addresses and their lengths in bytes.

Once the parts are all in memory you are now ready to save the linked program to disk. entering a question mark symbol will display the same information as the asterisk gave but an input prompt for a save name for the program will now be displayed. Make sure that the save disk is in the drive and enter the correct name that you want the program to be called. When you press RETURN, a second prompt will appear which requires the boot SYS address of the finished program. This will automatically run the program after it has been relocated by the generated Linker program.

After saving, the computer should be turned off and then back on again. The newly saved version can now be tested. Load it back in from disk and type LIST. This will display the Basic boot line which is SYS2080. RUN the program and it should automatically reconstitute your original routine and execute it without any further interference.

Note, you cannot link Basic programs.

Reducing the Load

Linker is a convenience program which avoids the need to store a separate loader on the disk but apart from this no real memory saving is achieved. For disk and memory economy you will have to put the newly generated Linker program through the Cruncher.

The idea behind Cruncher is to compress the memory used by any program, especially those using user defined characters and sprites.

When defining characters there are often large blocks of memory which repeat the same byte over and over again. Cruncher detects

these areas and stores the information as a single byte representing the character and another for the number of repetitions of this byte when the program is saved a decompression program is stored to stretch the program back to its original length.

When Cruncher is loaded, the screen display and commands are virtually identical to the Linker program but the upwards arrow, astensk and question mark commands behave in a slightly different way.

The upwards arrow returns the program to Basic This should be used at the end of each use of the Cruncher so that the program can be tested. It also restores all of the program parameters if the wrong program has been loaded for compression

To use this utility load the program which you want to compress by typing in the program name If you want to reassure yourself that it's in memory, enter the asterisk command The filename will be displayed alongside its start address ($0801 for linked programs) and the program length in bytes. A third column headed Crunch Length ought to have a zero byte at this point

To crunch and save the program, the question mark is used as before This time the program will appear to pause for a while as the memory is scanned and compressed and then the display will appear as for the astensk command but this time the crunch length will contain a value Some programs may show no improvement from this process and may even be longer because of the decompression routine! In such a case press the reset key (upwards arrow) and forget about compression

In most cases the number of bytes will be substantially fewer than before so a new filename can be entered and the start address prompt will appear This time the boot address for the Linker program should be entered ($0820) NOT the normal boot start for the program Remember that it is the linked version of the program which has been compressed and not the original program so it will need to be redistributed through memory after being decompressed

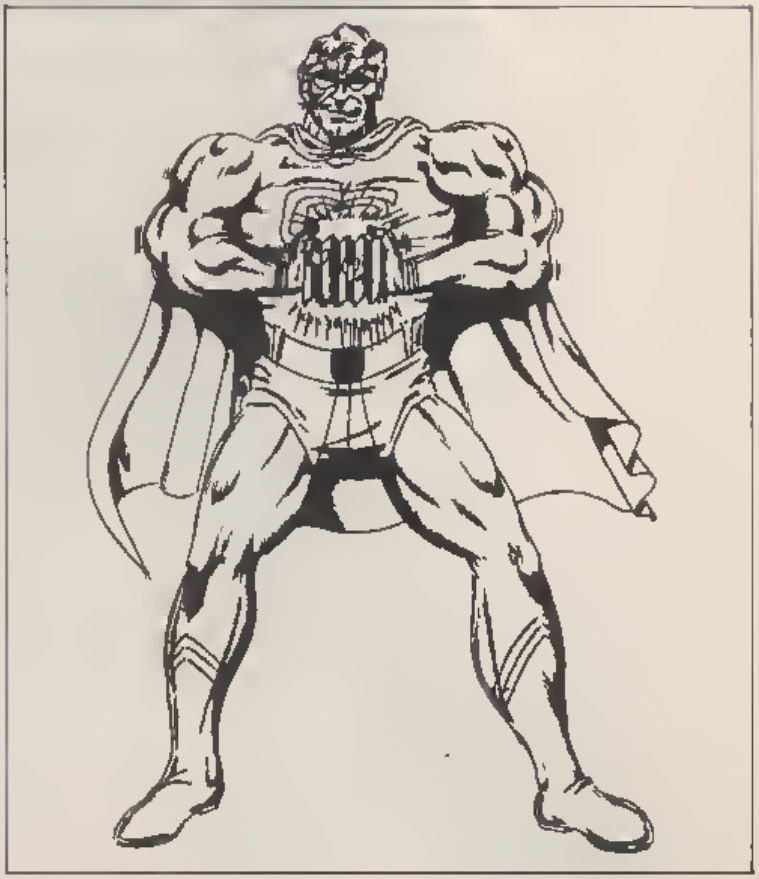
The program will then save and you can compare the compressed program's block allocation compared to the linked one to see the storage saving

The two utilities can be used independently of one another. It would be pointless to put a one part program through the Linker or to save a program which is only compressed by one or two bytes The start address for an unlinked program is no longer $0820 but the actual SYS address for the original, uncompressed program

The Linker and Cruncher programs can vastly improve your program storage and, used wisely, you can effectively increase the capacity of your disks.

Program commands

←	device number toggle (8/9)
@	disk status
@(command)	disk command
$	display directory
^	reset
*	display files in memory
?	save program to disk



Psymon

Get to grips with the machine code with this easy-to-use monitor.

By Gary Saunders

Psymon is a machine code monitor program for use with the C64. There are two versions here, one located at 36864, and the other at 49152. This is because you may want to use the area of memory where the monitor is actually located, whereby the second version will come in useful. Some assemblers also operate on this principle. Psymon has a total of 20 commands available to the user, listed below, together with examples of how they can be used.

Psymon commands

R-REGISTER prints register contents in the following format:
ADDR SP AR XR YR NV - BDIZC
Any of the values can be changed by simply cursoring up to the line, changing the required values and then pressing Return.

M-MEMORY Format: M Address 1 Address 2
This will disassemble all the memory contents in hexadecimal form from Address 1 to Address 2. Examples:
M 8000 9000
M 6000

G-GOTO Format: G Address
Jumps to the address following the G instruction. Example:
G FCE2

X-EXIT Format: X (Return)
This exits from the monitor to [illegible] SYS8 or a SYS to any location holding [illegible] bring you back to the monitor.

L-LOAD Format: L"FILENAME",DEVICE,[1]
Loads a program from disk (DEVICE=8) or tape (DEVICE=1) in binary form to the place it was originally saved from, relocated loads are also supported by [illegible] and examples:
L"TEST",8
L"SCREEN",[illegible] load)

S-SAVE Format: S"FILENAME", DEVICE, START ADDRESS, END ADDRESS+1
Saves a program to disk (DEVICE=8) or tape (DEVICE=1) with the appropriate start and end addresses specified. Examples:
S"MONITOR",8,9000,A000
S"MONITOR",1,C000,D000

V-VERIFY Format: V"FILENAME", DEVICE
This will verify a program from disk (DEVICE=8) or tape (DEVICE=1) with a program in memory and will report if any errors exist. Example:
V"TEST",1

H-HUNT Format: Address 1 Address 2 bytes
This will search (hunt) in memory from Address 1 to Address 2 any bytes which resemble the bytes specified after the H instruction, and will display on screen the address at which the bytes are present. Examples:
H A000 B000 "BASIC"
H E000 F000 4C 63 A6

P-PRINT Format: P Address 1 Address 2 byte
Used to dump (print) hex memory to screen with variable width, making it very useful for printers. Address 1 and Address 2 define the block to be printed. Byte should be a value between 1 and 32, which defines the number of bytes to be printed on a line. Examples:
P C000 D000 0F
P 9000 9A00 05

B-BRANCH Format: B Address 1 Address 2
This calculates offsets for conditional branch instructions. Address 1 is the location of the offset value (i.e. the location in which the calculated value is to be placed). Address 2 is the destination of the branch. Examples:
B 1001 103D
(i.e. 1000 BNE 103D will then be D0 3B in hexadecimal form).

C-CALCULATIONS Format: C Value 1 Value 2
This command adds and subtracts Values 1 and 2 and prints the answers in hexadecimal form on the screen. Examples:
C 0008 0006
C 4000 1000

F-FILL Format: F Address 1 Address 2 byte
This fills memory from Address 1 to Address 2 with the byte specified. Examples:
F 0400 07E7 FF
F C000 D000 EA

T-TRANSFER Format: T Address 1 Address 2 Address 3
With this command, memory from Address 1 to Address 2 will be transferred to Address 3 onwards. Examples:
T 2000 4000 6000
T C000 D000 0400

D-DISASSEMBLE Format: D Address 1 Address 2
This disassembles memory from Address 1

to Address 2 as object code and standard 6510 mnemonics. Examples
D A000 B000
D C000
Disassembly is stopped by pressing the STOP key

A-ASSEMBLE Format. Address mnemonic data
The assemble command converts 6510 mnemonics and data into the correct form to be stored into memory. Labels and other features of true assemblers aren't accepted. Typing Return with no mnemonic instructions following the address, allows you to exit the A mode. Examples
A 2000 LDA #00
A 2002 STA D020
A 2004 RTS
A 2005
There's a read-back check in case Ram isn't there, try assembling at Hex A000 to see this. (Note: No dollar signs are used in the mnemonics as in other programs)

J-JSR(JUMPSUBROUTINE) Format J Address
This will do a JSR (Jump SubRoutine) to the address specified. It will check for a RTS within the program running, and if found, will return back to the monitor. Examples
J C000
J 4000

$-HEXADECIMAL TO DECIMAL CONVERSION Format $Value
This converts a hexadecimal value into its decimal equivalent and prints the answer on the screen. Example.
$9000
(Note: There is no space between dollar and value)

?-DECIMAL TO HEXADECIMAL CONVERSION Format. ? Value
This converts a decimal value into its hexadecimal equivalent and prints the answer on the screen. Example
? 49152

↑-EDIT OBJECT CODE Format ↑ Address Object code Mnemonic data
Used to change the values of the object code of a disassembly. Use the Psymon (Hi) version for the following example.
The value 20 is to be changed to 4C at location C000, so D C000 C00D will print the disassembly needed. Now change line C000 to the following using the cursor keys:
↑ C000 4C 44 E5 JSR E544
and press Return to see the effect. The JSR should now have changed to JMP. Please remember to enter the original value (20) once you have finished this example

←EDIT MNEMONIC CODE Format
← Address Object code Mnemonic data
This uses the same format as with the object code edit, except it changes the mnemonic data of a disassembly. Example
disassemble from C000 as before, and now change to the following
← C000 20 44 E5 JMP E544
and press Return to see the effect. The 20 should now have changed to 4C. As before, remember to replace 20 back at C000

Error messages

Psymon displays both STATUS (ST) and I/O values instead of the standard error messages used in Basic. For the meaning of these values, look at Page 85 of the Programmer's Reference Guide. Any good disk drive book should have I/O values and their meanings

Getting the program in

The files can then be loaded and started as follows
PSYMON (LO)
LOAD"PSYMON (LO)",8,1
and then SYS 36864
PSYMON (HI)
LOAD"PSYMON (HI)",8,1
and then SYS 49152
To distinguish between the two versions, Psymon (LO) has a red background whilst Psymon (HI) has a grey background

Contributions

Written some programs? Got some programming wisdom to pass on? Or do you want to write about your own fields of interest? We're waiting for your contributions.

Commodore Disk User doesn't just offer you the chance of appearing in print, but of putting your programs on our disk for all to admire. We're always on the lookout for new programs for the disk. Anything goes, utilities, games or business programs in Basic or machine code - if we think it's good, we may well publish it.

Even if you haven't got a program to send, we'd love to pick your brains. If you have a field of expertise you'd like to explain or any tips and hints of interest to disk users, send them in.

But how do you go about preparing a submission? Just follow the guidelines and all should go well. You don't have to be a great novelist to contribute, but if you follow our simple rules then it will make our job a lot easier.

1) If possible all material sent to the magazine should be typed or printed out on a computer printer.
2) All text should be double-spaced, i.e. there should be a blank line between each line of text. You should also leave a margin of at least 10 characters on each side of the text.
3) On the first page you should put the following:
   Name of the article
   Machine that it is for (C64/128)
   Any extras required - disk, printer, add-ons etc.
   Your name
   Your address
   Your telephone number
4) The top of every page should have the following information on it:
   Abbreviation of the article title
   Your name
   The page number
   For example, suppose you had submitted a piece on C64 3D graphics. You should put something like this at the head of the page:
   3D/G Brown/1
5) Please make sure that you do not make any additional marks on your text, especially underlining.
6) Try to write in clear concise English. Your contribution does not have to be a great work of literature, but it must be comprehensible.
7) On the bottom of each page you should put the word MORE if there are more pages to the article, or ENDS if it is the last page.
8) If possible, enclose a listing of all programs.
9) Use a paperclip to hold the pages together. Do not staple them.
10) When submitting programs for the disk, submitting the program alone is not enough. Please tell us how to load, run and use it, preferably in as much detail as possible. If there are any interesting programming point involved, explain them to us.
11) Please do not submit machine-code programs as Basic loaders of the sort certain other magazines would accept. If you have any points, however, to make about the working of the program, an assembler source file on the disk would be handy, preferably for Your Commodore's Speedy Assembler.
12) Programs for the disk should be in as few chunks as possible. This makes our disk menu easier to set up.
13) Programs under 10 lines can be included in the text. If your program is longer than this it must be on a disk.
14) If your article needs any artwork, then supply clear examples of what you want. We don't expect you to be an artist, but we do need to see what is required.
15) Photos, if necessary, must be either black and white prints or colour slides. We can take shots ourselves, so don't worry about this too much.
16) Submissions of any length are welcome. A five-line routine may be just as welcome as a six-part series of 2000-word articles.
17) Payment varies quite a lot and depends on quite a number of factors, such as complexity and presentation of program. For articles, the number of magazine pages taken up is the salient factor.
18) All payments are made in the month that the magazine containing your article has appeared in print.
19) If we do find your submission suitable for inclusion in the magazine, we will write to you giving the terms of publication, the rate of payment, and an agreement form. Prompt return of this form will allow us to use your program as soon as possible.
20) If you want the program to be returned to you, should we find it suitable for publication, then you should enclose a stamped addressed envelope.
21) If you use a wordprocessor, then enclose a copy of your text on the disk and state clearly which wordprocessor you use.
22) Send your programs and articles to:
   Commodore Disk User
   Submissions
   1 Golden Square
   London W1R 3AB
23) Commodore Disk User cannot accept any liability for items sent to the magazine.

Disk Librarian II

More on the superb disk-filing system featured in our second issue

By Burghard-Henry Lehmann

The last time I gave you Disk Librarian, a program to help you build the files on your disks into a library. Now I give you Disk Librarian - the expanded version. The main concept of the program is the same, but I have developed it further and added new facilities.

But before I give you a thorough description of all the facilities of Disk Librarian and how to use them, let me briefly recapture the gist of the program:

What Disk Librarian does

Disk Librarian allows you to order the files on your disks into two large databases.

1. The Chronological File which holds the files contained on each of your disks in the way you build up your disk collection. With other words, a collection of all the directories of your disks.
2. The Categories File which lets you sort the files on your disks in a logical order, by categories and sub-categories. It's similar to what happens in your local library. The books are sorted by subjects, like language, psychology, transport, computers, geography, physics and so on, and each subject in turn is sorted into sub-subjects.

The Chronological File tells you which files your disk library contains, disk by disk. The Categories File sorts those same files into the logical order which suits your needs and interests.

In order to manage those two files Disk Librarian builds up a third file: the Master File. This is vital for the internal workings of the program. It contains all the names of the categories and sub-categories you have chosen. It also contains all the disk names and disk identity numbers of the disks you have included in your library and the date when you did so.

The Master File also tells the program if something has been saved or if that category is still empty.

Program modules

Disk Librarian consists of three program modules or parts.

Part 1 lets you set up your library and add to it.

Part 2 prints the contents of the Chronological File and the Categories File onto the screen and allows you to do some processing with each category or sub-category file, like sorting it alphabetically, moving files into an order which suits you more and deleting files.

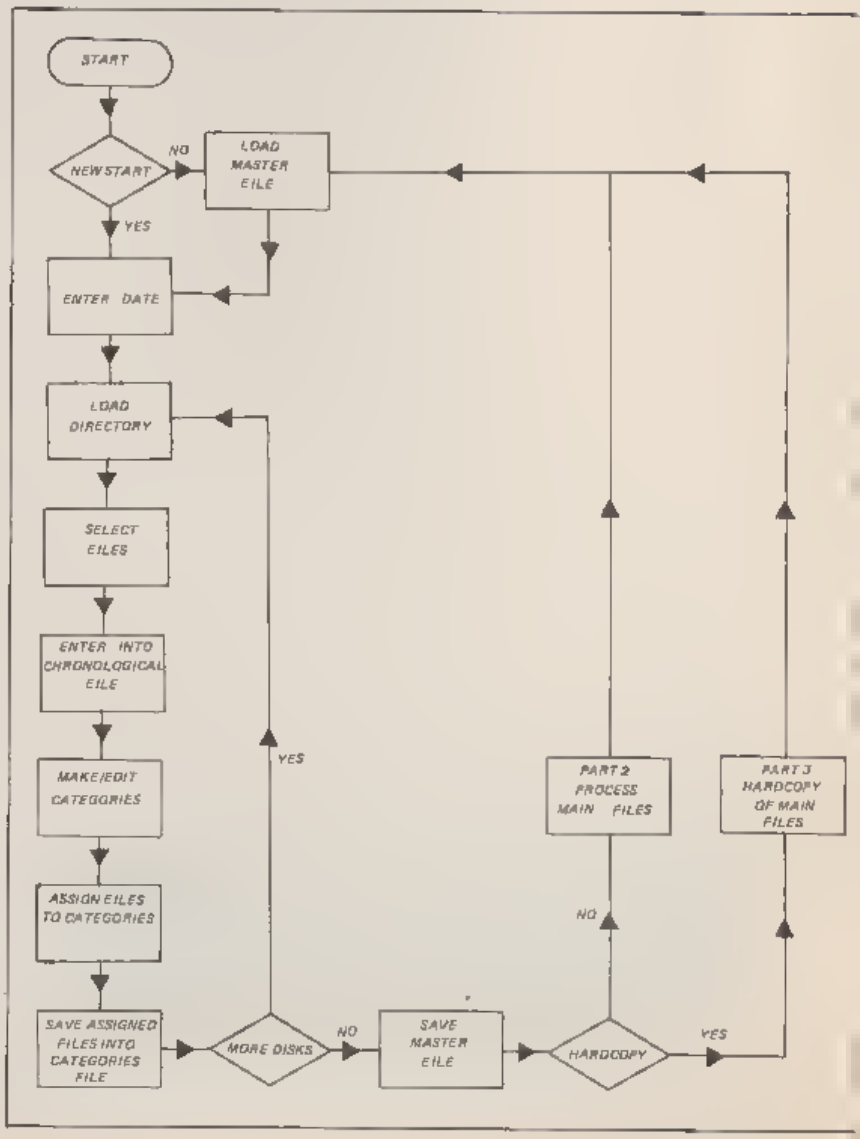
Part 3, finally, gives you a hard copy of the contents of the Chronological File and the Categories File.

How to Use Disk Librarian

The program is very easy to use: You enter Part 1 of Disk Librarian and first enter the present date.

Then you load the first directory of your disk collection.

Next you select the files in that directory which you want to be included in your disk library. (Many programs consist of a loader



program and then several files containing the program itself. In order to show that this program is in your library you need to include only one of those files in your library.)

The next step is that you start building up the Chronological File by getting the program to enter all the selected files of your first directory into the Chronological File.

Then you set up the Categories File by making a list of all the categories and sub-categories under which you want to order the files on your disks. For example, category 1: Textfiles, sub-category 1: Articles, sub-category 2: Letters, sub-category 3: My New Novel, and so on.

Next you assign each of the files you have selected from your first directory to the category or sub-category under which you want that file to be sorted.

Finally, you get Disk Librarian to save the assigned files under the correct category.

This you repeat with all the disks in your collection. Of course, you don't have to write the list of categories in each time, because this stays in the system. But you can at any given time add new categories or sub-categories to your list.

When you are finished with all your disks, the most important step is to save the Master File. As a matter of fact, to be on the safe side, it is a good idea to save the Master File several times before you are completely finished. You know, power cuts are not an unknown thing, and once you lose the Master File, there is no other way but to start all over again!

DISK LIBRARIAN - Facilities

Part 1 - Set Up and Update Library

Initial Menu: When entering Part 1 from program start, you have two options. You can load the Master File (function key 1), if you want to update the library you have already started, or you can start completely from scratch (function key 3). After both options you will be prompted to enter the current date, before the Main Menu of Part 1 appears on the screen.

Additional information given on the Main Menu Screen: In the top left hand corner the current ID number is displayed, that is the disk ID which would be used, if you should choose to use it (see facility "N").

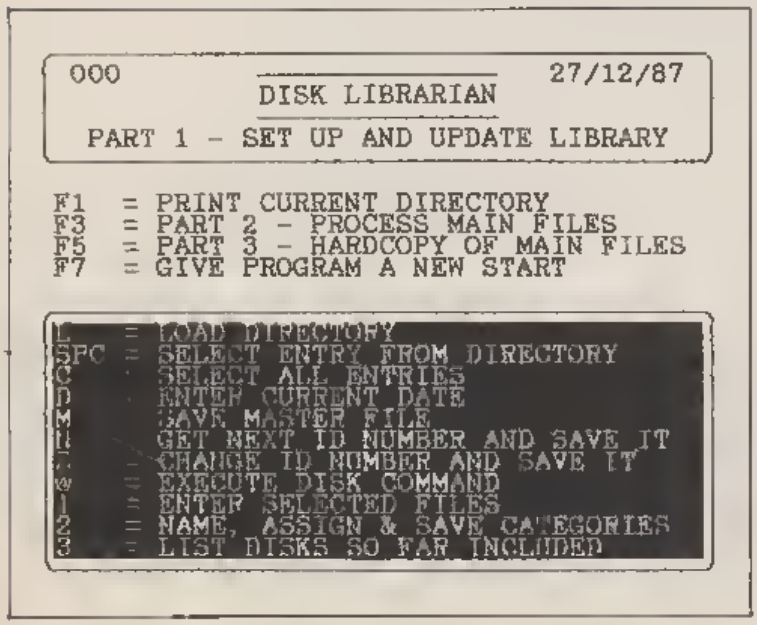
In the top right hand corner the current date is displayed, as you enter it, when prompted at the beginning of the program (see also facility "D", to change the date).

Function key 1: Print Current Directory. This prints the directory which has been loaded. You can toggle between this printout and the printout of the main menu simply by pressing function key 1.

Function key 3: Part 2 - Process Main Files. If you choose this option the second module of Disk Librarian will be loaded. Before that you will be asked if you want to save the Master File. If you have done any work with Part 1, always use this to save the Master File!

Function key 5: Part 3 - Hardcopy of Main Files. This works as above, only Part 3 of Disk Librarian is loaded.

Function key 7: Give Program a New Start. This is useful if you want to make a complete new start with Part 1. The program will revert to the initial menu of Part 1, giving you the choice to load the Master File (useful, if you had several versions made up with Disk Librarian) - function key 1 - or start completely new - function key 3.



L: Load Directory. This lets you load a directory from a disk. It will then immediately be displayed, so that you can select files or return to the Main Menu of Part 1 by pressing function key 1.

SPC (spacebar): Select Entry from Directory. Move the cursor to the file you want to select (you can move the cursor in all directions), then press the spacebar. The entry will be highlighted, which means, it has been selected. If you want to un-select it, simply press the spacebar again.

C: Select All Entries. This selects - or un-selects - all the files in a directory. Useful if you want the majority of files selected. Simply press "C" and then un-select the files you don't want with the spacebar.

D: Enter Current Date. This allows you to change the current date. You are prompted to enter the day, then the month, and finally the last two digits of the year. Always enter single digits with a trailing zero. E.g. 01/01/88, which stands for the 1st of January 1988.

M: Save Master File. This allows you to save the Master File at any given time. The Master File is always saved under the filename "Libr Master". This is important, because when loading the Master File, the program expects to find this filename!

N: Get Next ID Number and Save It. If you choose this option, Disk Librarian will tell you the ID number of the disk currently in the drive, the ID number it would give that disk, if you allow it to, and then give you the choice to

install that ID number on the disks or not. Afterwards the ID number in the system will be incremented, so that the next ID number will be one higher than the former one. This allows you to number your disks from 000 to 999.

S: Change ID Number and Save it. If you prefer your own system of ID numbers, you can use this option to change the ID number of the disk currently in the drive. In this case Disk Librarian lets you enter a five digit ID number, but remember, for its own purposes Disk Librarian looks only at the first three digits of any ID number. So, for Disk Librarian the first three digits of all the ID numbers of your disk have to be different!

@: Execute Disk Command. This lets yu you execute any of the standard DOS disk commans, like scratch, rename, validate etc. For example, "S:BLOGGS" would scratch a file called "BLOGGS". If you enter "$", the directory of the disk in the drive will be listed in the more usual fashion, as would be listed from Basic.

1: Enter Selected File. This enters all the selected files of a directory into the Chronological File. For this have the disk with the Chronological File in the drive. If it tells you that the disk has already been entered into the Chronological File, this means that it has found the first three digits of the ID number in the Master File. If you are sure that you haven't entered that disk into the Chronological File, change the ID number of the disk with the "N" or "S" facility, then try again.

2: Name, Assign and Save Categories: This facility lets you make up the list of categories and sub-categories you want, assign a category to a selected file and save the assigned files into the Categories File. When entering this facility, you can start straight away to enter names of categories and sub-categories. You are always prompted with the next category. If you want to switch from a sub-category to a main category, press return again, after you have been prompted with a sub-category. Use function key 7 to quit the process. Now you have several choices. "E" puts you back into the editing mode, "A" lets you assign files to categories, "S" saves assigned categories, and the Run/Stop key returns you to the Main Menu of Part 1.

3: List Disks So Far Included. This lists all the disks (ID number, disk names and dates) you have so far included into the Chronological File. Use cursor up and down to go through the listing.

Part 2 - Process Main Files

Initial Menu: Function key 1 loads the Master File and function key 3 gets you straight into the Main Menu, if the Master File is already in memory. You can choose this option, if you are coming from Part 3 of Disk Librarian.

Main Menu

Function key 1: Part 1 - Set Up and Update Library. This lets you load Part 1 of Disk Librarian. When the module has been loaded you will be asked first of all to load the Master File. This is necessary because the Master File is located at a different position in Part 1 than in Part 2 and 3.

Function key 5: Part 3 - Hardcopy of Main Files. Loads that module.

Function key 7: Give Program a New Start. This returns you to the initial menu of Part 2, from where you can load the Master File again.

@: Execute Disk Command. Exactly as in Part 1.

1: List Chronological File. First the disk names, ID numbers and dates are listed to let you choose the disk you want to look at. Use cursor up and down to select the disk you want, then press return. Now the content of that disk, as entered into the Chronological File, will be listed on screen. If there are more files than fit onto the screen, use cursor up, down, left and right to bring the rest of the files into view.

2: List Categories File. First your list of chosen categories and sub-categories is printed on the screen. Again, use cursor up and down to find the category whose contents you want to be listed, then press return to select that category. A star on the left hand side shows you if any files have been saved under that category.

Once the contents of the category chosen have been listed, you may do a certain amount of editing to the file. "A" sorts the whole file alphabetically. "D" deletes the file next to the cursor. If you press the spacebar, the file next to the cursor will be highlighted. If you then move the cursor up or down to a required position and press "M", the file you have highlighted will be moved to that position. In this way you can order the files within a category by hand in whatever way you want.

Run/Stop will return you to the list of categories and sub-categories. If you have changed the content of the category by using any of the facilities described above, you will be asked if you want to save the modified file.

Part 3 - Hard copy of Main Files

Initial Menu: This is the same as in part 2.

Main Menu:

The function keys do very much the same as in the other modules.

@: Execute Disk Commands. As in the other modules.

1: Printout of Chronological File. This sends the whole of the Chronological File, disk by disk, to the printer. Press Run/Stop to abort the printing.

2: Printout of Categories File. This sends the contents of all the categories and sub-categories under which files have been saved to the printer. Again, Run/Stop aborts the printout prematurely.

3: Printout of Disks Included. This sends the same list as in Part 1, option "3" to the printer.

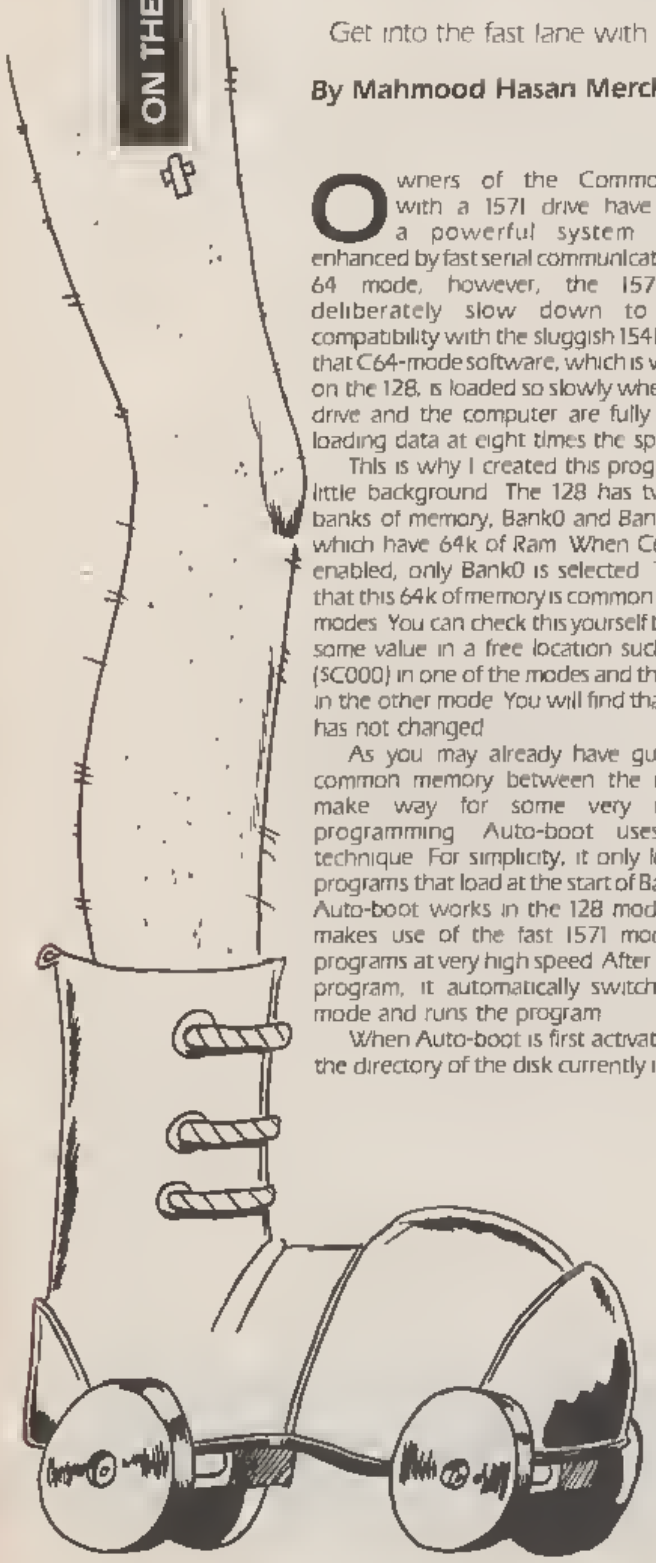
4: Printout of Category Names. This sends the same list as in Part 2, option "2" to the printer.

ON THE DISK

C128 Auto-boot

Get into the fast lane with this speedy C128 loader

By Mahmood Hasan Merchant



Owners of the Commodore 128 with a 1571 drive have access to a powerful system which is enhanced by fast serial communications. In the 64 mode, however, the 1571 has to deliberately slow down to maintain compatibility with the sluggish 1541. It is a pity that C64-mode software, which is widely used on the 128, is loaded so slowly when both the drive and the computer are fully capable of loading data at eight times the speed.

This is why I created this program. First a little background. The 128 has two parallel banks of memory, Bank0 and Bank1, both of which have 64k of Ram. When C64 mode is enabled, only Bank0 is selected. This means that this 64k of memory is common to both the modes. You can check this yourself by POKEing some value in a free location such as 49152 ($C000) in one of the modes and then peeking in the other mode. You will find that the value has not changed.

As you may already have guessed, this common memory between the modes can make way for some very interesting programming. Auto-boot uses such a technique. For simplicity, it only loads single programs that load at the start of Basic ($0801). Auto-boot works in the 128 mode, hence it makes use of the fast 1571 mode to load programs at very high speed. After loading the program, it automatically switches to C64 mode and runs the program.

When Auto-boot is first activated, it reads the directory of the disk currently in the drive, and lists a maximum of twenty-six programs (it ignores other filetypes). Press the relevant key and the program is loaded at the high speed of the 1571. After the load, the drive is set to 1541-mode and the 64-mode enabled. The program is automatically run. To load Auto-boot, type LOAD "AUTO-BOOT",8 and RUN.

I have designed Auto-boot in such a way that it allows very long programs that start with a Basic line to work. I have tried loading a program exceeding 200 disk blocks in length with it and it worked perfectly.

Some tips. Even after loading a program using Auto-boot and then loading other programs while staying in C64 mode, you can get the original program back in memory with the following procedure: Press the RESET key keeping the RUN/STOP key depressed. This would enter you into the 128 ML Monitor. Now enter.

G !0400

to activate the code at $0400, Bank1. After 2 or 3 seconds, the 64 mode will be enabled and the program run.

You can generate a SYS file using Auto-boot as the first program on a disk on which you can then save other C64 programs. Whenever you want to load a program, enter the disk and press SHIFT-RUN/STOP.

Another option could be to generate a Binary file and create a BOOT sector on the disk that loads and activates Auto-boot. This would be very convenient. Just insert the disk and turn on the 128.

Level-headed Thoughts

It's amazing! Multilevel games seem to fit a quart into a pint pot. How do the professionals do it?

by Norman Doyle

Every screen of a multi-level game has a maximum of 1000 characters and each character needs a colour. This means that 2000 bytes have to be supplied for each new screen displayed. Even assuming that the maximum 64k Ram is available for storing the screen map information, a game could only have a maximum of 32 screens!

Such a system would leave about 1536 bytes for game control routines, sprite and character definitions and the screen display itself. So how do programmers produce multi-level games of such diversity?

Basically, it's a trade-off between sensible graphic design and cunning program compression. The fewer characters that can be used, the more memory remains free for other purposes. By recombining the same graphic set in different ways, a considerable amount of memory can be saved. More economies can be made by thinking carefully about how the screen information can be packed into memory in groups rather than as single bytes.

To understand all of the methods used would fill several issues of Commodore Disk User so this article will only attempt to provide the key which will enable you to go on to explore this intriguing world on your own.

Block Building

If you consider each character position on the screen to be a 'block', then reducing the number of blocks would mean that the corresponding screen map would be smaller. For example, the 1000 block map could be reduced to just 60 blocks by defining each block as a four character by four matrix.

To do this requires extra code to construct and place the blocks but even allowing for this, the memory saving is still considerable.

From a single set of 255 redefined characters it is possible to construct more than 128 blocks but what is needed is a control system to distribute the blocks around the screen. For those who want to study the construction of character blocks the Shoot-'em-up Construction Kit on the Outlaw label by Palace Software is heavily recommended. If you have a copy of Tony Crowther's 3 Into 1 Plus program from the first Commodore Disk User disk you can use it to help construct the blocks and the control program assumes that something like this has been used.

Screen Memory

The control system for the construction of screens needs some kind of internal organisation. The character definitions have to be stored in some way. First we'll make the following assumptions: the redefined characters are stored from $0800 upwards, the block definitions are stored from $1000 upwards and the screen lies at $0400.

The block definitions are stored just as though they were layed out on a screen measuring 40 characters wide by 52 characters high. This means that each row of the first character block will be found at $1000, $1028, $1050, $1078.

Storing the block information for each screen requires sequences of 60 blocks from $1800. If we extend this to 64 we lose four bytes per screen but this can be utilised for storing parameters which refer to that particular level of the game. The advantage of using 64 is that it is a handy number for manipulation.

How does the memory saving help in real terms? Ignoring the screen maps, $0000 to $1800 is used for the screen display, characters, blocks and Basic control locations. Most of the Basic control area can be used for the game control program but we shall ignore this completely for the moment.

To make things even harder, let's assume that the top of memory Rom is still required, which loses a further 8192 bytes. Sprite, sound and screen colour information is stored in a further 4096 bytes and the memory maps for 32 different screens will now occupy 2048 bytes instead of the original 64,000.

How much memory is free now? Unbelievably, there is still 47k available! Even if the screen colour information is stored as the full 1000 bytes per screen, there's still well over 10k for the game itself and sprite definitions.

Transferring to Screen

Each screen map will contain the number of a particular block from 0 to 127. To find the start of each block a jump table can be set up for the top left byte of each block. Similarly, another jump table can be set up for the 60 screen locations. For the first screen the routine would resemble the following listing:

Once the player has completed the first

level, the pointer at $FA is modified to point to the start of the second screen's map and the routine is called again and so on through the levels

Colour Compression

Few program screens use a different colour for each character square and sensible economies can save memory The most extreme saving can be obtained if only four colours are used over the whole screen Then Multicolour 1 and 2 can remain the same once set for that level, similarly the background colour is fixed Now to set the fourth colour a simple memory fill routine can be used to poke the value into $D800 to $DBE7, the screen colour memory

Where can we store these four pieces of information? If you recall there were four bytes 'wasted' when the screen block memory maps were created, we need four bytes and here they are!

This results in a substantial saving of memory, leaving room for alternative character sets, gameplay routines and sprites A smaller but significant saving can be made in more ambitious game screens by designating different colour information for each character block using a system similar to that outlined for the character blocks themselves

The Way Ahead

Once the idea of memory-saving routines has been mastered many other ideas should occur to a good programmer. Try limiting the size of the screen by having a generously sized score panel which looks the same on all levels. Use just the bottom area of the screen for defined characters and leave the rest blank to represent a clear sky or create a tunnel by using strips at the top and bottom of the screen

Scrolling screens can be produced by using extended maps The start screen is poked into place as we have seen and then a series of registers can be set up to read a scroll map block by block When the characters are located each of the four rows or columns are poked to the edge of the screen ready to scroll

There are many ways to store screens and the occupied memory space can be further limited by using just fifteen blocks per screen Now one byte of the block map can be used to store the numerical values of two blocks In hex the block number will lie between $00 and $0F As you can see one half of the number is always zero so another 0-15 value can be stored instead of the zero The control system can split the byte into two components by simply ANDing the value with $F0 followed by four ASL commands if the high part of the byte is needed The low part is easily obtained by ANDing with $0F

Now it's up to you, the methods used have been modified by many professional programmers to suit different requirements of gameplay All the basics are here - all you need add is your own ingenuity

Monitoring Monitors

Although monitors may offer slightly different facilities they all do the same basic job but what is that?

By Norman Doyle

If you want to explore the computer's memory it can be done from Basic by peeking into various locations. This is a long and laborious job which can be more easily achieved by using a monitor.

At their lowest level, monitors are a collection of simple machine code programs tied together by their own single symbol keywords. Using these keywords machine code routines can be written, loaded, saved and examined with the minimum of fuss. This is different to an assembler in several distinct ways. An assembler is a full-blown programming language which simplifies the writing machine code routines. The principle advantage which this gives is that a location can be 'labelled' - given a name chosen by the user - and instead of remembering the value of a specific location it can be called up by name.

The disadvantages of using an assembler are that the routines backing up this operating system take up quite a lot of memory space and examination of the program code and testing of the routines is rarely possible without a lot of fuss.

This is where the humble monitor comes in useful. Once a program has been created using an assembler, it can be loaded through a monitor and then examined, tested, debugged and modified.

It is possible to write routines using a monitor and often this is the best option if it's just a short program. Longer projects require so much note taking with a monitor that most programmers soon resort to an assembler for speed and ease.

Apart from loading and saving, the routines which most monitors have in common are mnemonic assembly and memory dumping, moving, modifying, byte searching and block filling.

Mnemonic assembly or disassembly is when the three letter commands (LDA, STA, BEQ etc.) are used instead of their decimal or hexadecimal equivalents. This makes the job easier because a disassembled program can be readily understood without the need for referring to a manual every few bytes. For example,

```
LDA #$20
STA #0400 CRA1
RTS
```

is a lot more meaningful than $A9, $20, $8D, $00, $04, $60.

A typical monitor would give both forms of notation on a single line after the memory address of the first byte of each command.

```
*C000 A9 29          LDA #$20
*C002 8D 00 04       STA $0400 CRA2
*C005 60             RTS
```

In this particular monitor the code can be changed by typing over the numerical value or mnemonic letters. This is because the asterisk at the start of the line is a single symbol command to the operating system. Some monitors would require a different symbol for changing the mnemonics to the one which permits the numerical values to be changed.

Sometimes a block of memory is used for storing data for use by a coded routine. A disassembly would be fruitless so an ASCII block printout facility is usually available.

In the same way as values could be changed through the mnemonic disassembler, the values in the block display can be altered.

Icing the Cake

Without any other routines, these two commands would make quite a serviceable monitor if only a few minor changes needed to be made or the user merely wanted to inspect a ready written program. This is rarely the case and monitors are generally used for modifying and correcting small errors in advanced projects. Often this means adding extra code to cater for some unforseen circumstance, maybe a comparison filter fails to prevent a character from entering a print string or perhaps a routine omits to set or clear the high byte when a sprite moves across the screen. In either case it would probably be easier to experiment through the monitor before returning to the assembler. To do this it will often be necessary to move a block of memory to create a space for the extra code.

Monitors cater for this with a block move command which normally takes the form of a single character command, normally T for 'transfer', followed by the start and end locations of the block to be moved and then the location for the new resting place of the first byte:

```
T C000 C100 C300
```

One problem of moving blocks of code in this way is that some memory jumps accessing locations within the block that has moved will no longer be valid. Branches are unaffected

because they are relative to any position they occupy. In other words, branches are simply a jump of a number of bytes and if this is something like 20 bytes, the 20 bytes will lead to the same point in the code no matter where it is placed. The danger areas are JMPs and JSRs.

In the example above a branch at $C050 which pointed to $C03C would be placed at $C350 and point to $C33C after relocation. Unfortunately a JMP $C03C would still be pointing to $C03C after relocation and manual alterations would have to be made to correct it to point to $C33C.

Some monitors cater for this situation and will automatically change JMPs and JSRs when a transfer is made but the secondary problem with this occurs when data blocks within the code block exist. A monitor would make no distinction between JMP $C03C (ie bytes $4C, $3C, $C0) and a data string which may also be $4C, $3C, $C0. Often this is avoided by allowing the change to be made after relocation so that the user can alter only the code sections and leave the data areas alone.

Most monitors don't allow for these refinements because the idea is to keep the monitor as short as possible. Refinements use extra bytes and therefore limit the amount of memory free for code so a basic block move routine is all you get.

If this is the case some method of finding the JMPs and JSRs is needed. The command for this is normally H for 'hunt', and the format for the command is to give the start and end address for the search and then the hex value of the token you're looking for, in these cases $4C or $20. For JMPs this would be:

```
H C300 C400 4C
```

Most hunt commands will accept a string of characters for the search so if you want to find where a jump to print something to the screen is located (syntax: JSR $FFD2), the following command would find it:

```
H C300 C400 20 D2 FF
```

Another problem with moving code about is that the routines merely copy the code to the new position rather than moving it. This means that if location $C000 held $A9 before the relocation, it would still contain $A9 after the transfer. This can cause problems when writing in the additional code, it would be better if this area could be cleared with a zero byte or something equally as uniform. For this reason the memory fill command 'F' is used:

```
F C000 C2FF 00
```

After pressing the RETURN key each location up to $C2FF would be altered to a zero byte. When I use this command I often choose to fill with the hex value for the NOP command ($EA) because it means that testing the routine will automatically cause it to run through the null operations after executing the new code until the relocated code is found.

At some point you'll want to test the code and two kinds of execute commands are in common use. J and G.

If G C000 is used the code will be executed from $C000 until a BRK command is met, then the control will be handed back to the monitor.

If J is used before the address the code will be executed from that location until a RTS command is reached which does not relate to a JSR within the code under test. When this is reached the control once more returns to the monitor.

Whenever a monitor is entered whether by a SYS command from Basic or from a test routine, it is customary for a print out of the last address, the register contents and the status register to be printed out. This can be a handy debugging device and a register printout is often forced by the parameterless R command.

After creating some routines there has to be a save command which will allow the memory block to be saved to tape or disk. The syntax for disk is as follows:

```
S"progname",08,C000,C400
```

Some monitors allow the code to be saved for relocation when it is loaded again. For example, a tape program saved as follows would always reload at location $0200:

```
S"progname",01,C000,C400,0200
L"progname ",08
```

For a relocated load this may read:

```
L"progname",08,2000
```

Gilding the Lily

Most monitors go beyond these basic commands but this is where the types are subject to the whim of the programmer. If you read about the monitor included on the disk you'll see that extra commands allow HEX-DEC or DEC-HEX conversions to help those who are vexed by hex and a calculation feature. These are rarely found on monitors and are welcome additions.

Using a monitor efficiently means that you must memorise all of the commonly used commands and, as with most things, practice makes perfect.

Although our monitor fits into normal RAM space, the current trend is for monitors which reside in ghost RAM inside a cartridge. This means that all of the computer's memory can be accessed and, because program RAM isn't at a premium any more, it can be much more sophisticated. Perhaps future monitors will be developed to a point where labels can be used which will then make the old assembler programs redundant.

Disk dungeons

In which Grontol offers advice to and airs the views of benighted adventurers everywhere

Welcome to Grontol's pages. This issue features four reviews of games with a reasonably modern setting - not a dwarf or a dragon in sight. There seems to be a trend in traditional adventures away from a pure fantasy setting. These in turn have become the background for role-playing games.

The news, this issue, is somewhat thin on the ground. None of the four games reviewed here really make you sit up and take notice and you'll need to sit twiddling the thumbs waiting for the real goodies to arrive. For a start, Jinxter is the next game from Magnetic Scrolls, publisher of the excellent Pawn and Guild of Thieves. A vital game to mention is Ultima V which is supposed to be a lot bigger and better than Ultima IV, one of my favourite role-playing games. Finally Beyond Zork is a new concept from Infocom. Set in the lands of Zork well known to most adventurers, it is a cross between role-playing and traditional problem solving. Hopefully, all three will have arrived in time for an in depth review in the next issue.

Not a penny more, not a penny less

Until the advent of this game, the computer screen was one of the few remaining places where you could be certain of not coming across Jeffrey Archer in any shape or guise. Now, that barrier has fallen, as Domark have produced a game based on Mr Archer's first, semi-autobiographical novel, Not A Penny More, Not A Penny Less. When (if) he reads this review, I suspect that Mr. Archer may be less than keen on the next collaboration.

Not, I hasten to add, is this in any way the fault of Mr A. Whether you like him or not, it cannot be denied that, whereas he is never going to be in line for a Booker Prize, he can (and does) tell a rattling good yarn. No, the fault lies entirely in the game so please Mr A., no law suit, please. I can't afford half a million quid at the moment, certainly not on what the editor pays me.

The story is that you, Stephen Bradley - an Oxford Don - are one of four people that have been swindled out of one million dollars (early 1970's prices) in a con involving stocks and shares in a false oil company set up at the time of the North Sea Oil boom. You must persuade the other three characters that it is in your collective interests to find a way to get your money back.

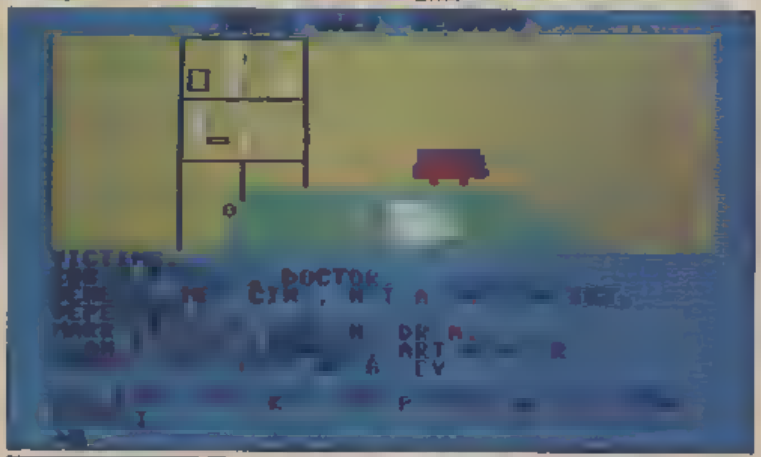
The problems with the game, and there are many, stem entirely from the game mechanics. The parser is execrable. Bugs abound. The graphics are poor. The speech is dire. Playability is almost non-existent. Apart from that, the game is fine!

Now that we are used to seeing parsers such as those used by Magnetic Scrolls, Infocom and Level 9, the one used here goes back to the Dark Ages. Directions must be typed in full i.e. you must type Go Northeast rather than just NE. Examining items may or may not give a response. Examining your dossier fails to reveal a vital phone number that only shows up when a policeman arrives at your door! And if you haven't shut your dossier before he arrives, then huge chunks of furniture in your room disappear! There is no inventory so that you do not know what your are carrying at any given time.

The game is totally linear in structure, the worst possible case for an adventure. This means that you cannot leave your room until you have read your dossier and made some phone calls (for added realism, phoning Scotland Yard takes you straight through to your stockbroker). Get the slightest thing wrong or out of sequence and it is Game Over time. If you get stuck on a particular problem, tough. There is nothing else for you to do in the meantime.

This game is a perfect example of everything that is bad in an adventure. At £16.95, it is grossly overpriced, even though it does contain a copy of the book. Buy the book by all means but save yourself the rest of the money.

GRH



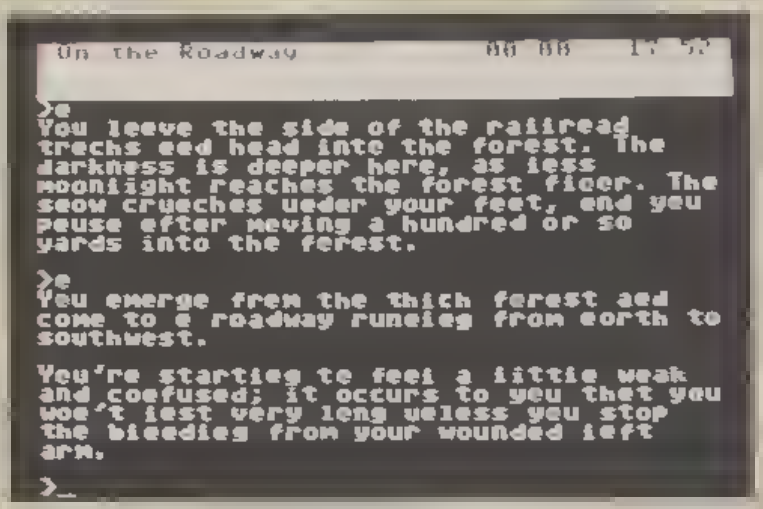
AT A GLANCE

Title: Not a Penny More, Not a Penny Less
Supplier: Domark, Domark House, 22 Hartfield Road, Wimbledon, London SW19 3TA. Tel: 01-947 5622
Price: £16.95
Graphics: Tedious block-fill routines
Sound: Mispronounced words coupled with hiss
Addictiveness: I won't load it again
Playability: You'll have more fun playing spot-the-bug

Borderzone

I spy with my little eye, something beginning with Y Give in? The answer is Yet Another Infocom game Spying is the name of the game, though, as you embark on a train journey from the East European country of Frobnia towards the border and the relative safety of Western Litzenburg

The story is in three parts showing how the same series of events affect three different people - an American businessman, an American agent and a Russian agent The story involves a plot to assassinate the American ambassador which you must do your best to thwart



The game runs in real time so that you must make decisions as quickly as possible or suffer the consequences of people carrying out their normal duties without your interference A typical example is a coded message that you are asked to deliver By the time you realise that you should reply in Frobnian and looked up the appropriate phrase in your codebook your contact has decided that you are the wrong person and moved on

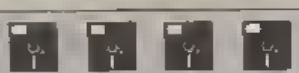
Everybody is extremely suspicious of everyone else, the police doubly so, and how they react to you depends on your actions and comments They may give you a clear run, follow you or simply arrest you So not only must you accomplish your mission, but also take steps to divert attention away from yourself

The game is beautifully packaged, complete with map, matches, train ticket and a wonderful guide/phrase book The parser is the usual Infocom one, which means very good, although it is beginning to look a little dated The absence of the Ram Save command is the most notable omission Saving to disk seems to take forever

There is an online help facility provided, should you get stuck, although I am not too keen on this for a couple of reasons Firstly it suggests that you might need it and so the temptation is there to use it It is like having the answers to a crossword printed on the same page as a puzzle Everyone says that they are not going to cheat, but, somehow, it never quite works out like that I would rather have the temptation removed completely. Secondly, if you do look up the list of hints (I only did this so that I could report back to you, you understand) you get a list of problems that need solving and this again tends to give the game away

That apart, the game is well up to Infocom s usual standards and fans won't be disappointed although the £25 might take some saving up for **GRH**

AT A GLANCE



Title: Borderzone
Supplier: Activision/Infocom, 23 Pond St, Hampstead, London NW3 2PN Tel 01-431 1101
Price: £24 95
Graphics: N/A
Sound: N/A
Addictiveness: Worth defecting for
Playability: A lot easier than leaving Frobnia

Deja Vu

This is the first of a series of adventures originally written for the Apple Macintosh and other sixteen-bit machines It was originally intended to be controlled by a mouse As it is, you have to make do with a joystick

You awake with a hangover that feels like nothing you have ever experienced before As your eyes gradually defy gravity, you find that you are inside a toilet stall with absolutely no recollection of how you got there Come to think of it (which you can t because it hurts too much), you don t actually have much of a clue as to who you are

As you make your way out of the cubicle and explore the rest of the rooms in the bar (for that is where you are) things are not improved by the discovery of a corpse You have the distinct impressive that you are going to be framed for this Or, as you can't remember anything, perhaps you really did do it - an old poster seems to indicate that you were/are a prize fighter so it would appear that you are no stranger to violence It would appear that you might use those talents as you try to escape the clutches of both the cops and the mob

Controlling the game takes a fair bit of getting used to There is no typing to do Everything is controlled via windows, icons and menus Executing a command involves selecting a verb from the menu, which offers a fairly limited choice and then pointing to whatever you want to act on in the main picture If you want to pick something up, you can try and drag it over to a box marked Inventory

The hardest part to get used to is the system of windows. Every time you open an object, for example your coat or your wallet, a window is opened showing the contents of whatever it was you opened. These windows all sit on top of each other and you have to get the hang of closing the right ones, moving objects from one to the other etc.

I found the joystick control somewhat inconvenient to use. Everything appeared to happen in slow motion and I longed for a proper mouse on a sixteen-bit system. Even typing seemed preferable. However, the game does give you a fair insight into how software is going to be developed in the future.

The game control is only a minor niggle, and I enjoyed playing the game. The atmosphere is extremely well-developed and the undercurrent of violence and crime is never far away. The game is good value at just under fifteen pounds and I look forward to the next games in the series. **GRH**

AT A GLANCE

Title: Deja Vu
Supplier: Mirrorsoft/Mindscape, Athene House, 66-73 Shoe Lane, London EC4P 4AB Tel: 01-377 4645
Price: £14.95
Graphics: Very well-drawn
Sound: N/A
Addictiveness: They made me an offer I couldn't refuse
Playability: Awkward to start with

Gunslinger

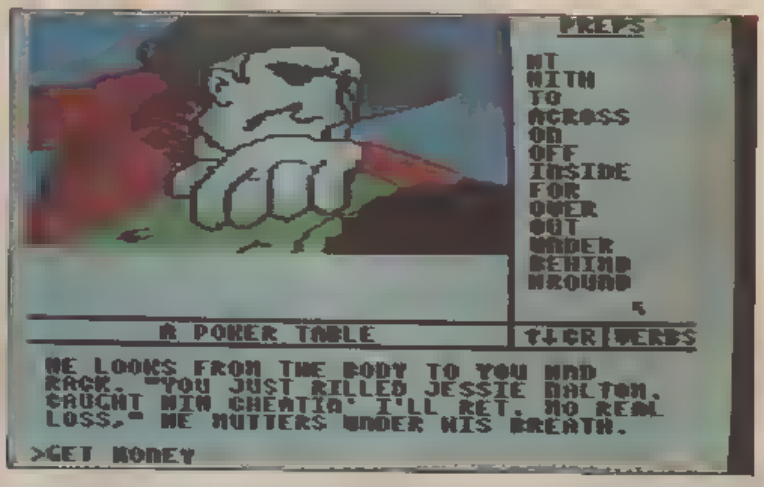
An adventure set back in the days when men were men, cattle and women were rustled, the goodies shot from the hip, got shot only in the thighs and shoulders (without any blood of course) and there was a plentiful supply of baddies and injuns just waiting to be shot. Yup, you've guessed it, Gunslinger is a good old fasioned western.

You are Kip Starr, a retired Texas Ranger who has just heard that his best friend is about to go dancing on the end of a rope down Mexico way. Naturally, you decide to go and rescue him (it wouldn't be much of adventure otherwise) and so must make your way down South avoiding all of the pitfalls mentioned above as well as all the other western clichés - ghost towns, deserted mine shafts and crooked gambling games ad infinitum.

It is a gambling game in fact that brings you into your first brush with the notorious Dalton gang. As you are holding AKQJ10 in Hearts, you are somewhat surprised to see your opponent try to claim the pot (all of 87 cents) with four aces. The Sheriff wants you out of town, after you kill the offender, and your share of the winnings is just enough to buy you a ticket on the next stage. Which just happens to be ambushed by injuns leaving you alone in the middle of a desert.

The game can be controlled entirely via a joystick (although you can still use the keyboard if you prefer). On the righthand side of the screen is a list of all the possible verbs you can use (about five pages worth) and you select the one you want from the menu. This is followed by a list of nouns, then prepositions and finally nouns again. This means that whereas most commands can be entered as simple verb/noun structures, you can be more specific as the occasion demands for example KILL MAN will assume you are using your hands. KILL MAN WITH GUN usually has the desired effect (you are the goody, after all, so you can't get shot too often). Moving off the list of actions brings up a compass allowing you to move easily.

The game is a large one (two double-sided disks) but quite a lot of the available memory is used for the large illustrations. These are of good quality, although nowhere near as good as those used by Magnetic Scrolls. The game itself plays fairly well and will definitely appeal to would-be Wyatt Earps. **GRH**



AT A GLANCE

Title: Gunslinger
Supplier: Datasoft
Graphics: As colourful as the wild west was
Sound: N/A
Addictiveness: I'll keep loading my six-shooter
Playability: A lot better than that 5-Ace deck of cards

Epyx

Epyx is one of the most successful US software houses infiltrating the UK. Tony Hetherington tells the Epyx epic

California Games

A succession of hit games such as Winter Games, and now California Games, has established Epyx as one of the top US software houses. Unlike Electronic Arts and Microprose who have travelled across the pond, Epyx has stuck with its licensing deal with US Gold. The partnership has resulted in countless number one hits. In these pages we look at the past classics that are still available, two current hits and the future hits of '88.

The Games series of games has been the cornerstone of Epyx's success that started with the Olympic style Summer Games. This appeared just after the decathlon bubble burst and so didn't get the success it deserved. However, Summer Games II followed shortly after and established the Games format. One to eight players could represent the countries of their choice in a series of events that could stand as individual games themselves. After each event was over, the winner was rewarded with a medal ceremony and the games were opened and closed with graphic displays such as the lighting of the Olympic flame or a fireworks display.

Summer Games II moved away from the established Olympic events and included challenges such as canoe slalom and horse jumping events.

The third in the series is Winter Games, which still ranks among my favourite games. This time the six events are played against a snow-covered background and you can almost feel the cold as you set off down the ski jump ramp, leap through acrobatics in the hog dog aerial, hurtle down the bobsleigh run, shoot and ski your way across the cross-country biathlon course and enter the rink for figure and freestyle skating.

World Games takes its players around the globe in a series of events that test your barrel-jumping nerve in Germany, your cliff-diving skills in Acapulco, rodeo-riding in America, Caber Tossing in Scotland (gasp!). After 8000 miles and seven gruelling events you end your tour stomach to stomach with a Japanese Sumo Wrestler.

The latest in the series takes you to the sunshine, sea and surf of California for six scorching events to save the honour of yourself and your sponsor. Sponsored computer sports have arrived!

To win the tournament you will have to ride the surf long after the others have wiped out, throw and catch a frisbee further than ever before, run the Gauntlet of the highest BMX course you'll ever ride, show off your soccer skills with a footbag, ride your skateboard in the half pipe while performing crucial moves and finally, rollerskate through all the obstacles laid before you to victory.

Each and every one of these is highly recommended which must make the Games games the most successful series in software history. All but Summer Games are still available individually but you can still tell from your Olympic prowess as Summer Games is just one of the four games in the compilation Epyx Epics.

If you pick up a copy of Epyx Epics (£9.99) you can grab yourself a chunk of computer history as you wager Pitstop II which was the first split-screen racing game. Impossible Music and Breakdance.

Super Cycle

Impossible Mission swept the awards in 1985. It combined the best in platform action with digitised speech and laughter from the evil Elvin Atombender and his deadly and intelligent killer robots. Your mission is to enter Elvin's secret hideout. Armed with only your wits and a pocket computer you must find and disable the control centre and foil his plot to destroy the world. To succeed you have to combine athletic skill to navigate the platforms of each room with the brain power to solve the puzzles posed in the form of computer codes and the deadly shape laser-firing robots.

Pitstop II became the definitive racing game as split screen action brought player against player for the first time. Driving faster than the other cars is only part of the problem. A good pit crew (player-controlled) who can change worn tyres and top up fuel in a matter of seconds can win the day.

Two-wheeled racers who wanted arcade-style action climbed onboard the Super Cycle in their droves and steered it quickly to number one. This bike racing game gives you course after course of new terrain to beat in a race against the other riders and the clock.

The latest sports simulation from Epyx is also the first in a new series of games. Street Sports Baseball is baseball as it is actually played by those (most people) who can't play in stadiums like Shea Stadium, home of the New York Mets. Instead of astroturf kept neat by groundstaff you have to play in a park strewn with rubbish or on a carpark! Once the "stadium" is selected you must pick your team from the local kids including girls! Each player has their own abilities making some of the girls surprise winners. For example, Butch plays a little wildly but can catch superbly whereas Kim is a good base stealer but lacks concentration in the field.

Getting to know the strengths and weaknesses of the players is only half the game. A match can be decided because a fielder dropped a catch because they ran into a dustbin or tripped over a bottle or fell down a hole. Knowing the ballpark is essential and is helped by the game's split-screen display. The leftside of the screen is dominated by 3D view of the pitcher and batter, a crucial fielder or one of the bases. It's all backed up by a smaller top-down display that helps you to judge the angles and shows the position of the runners on the bases.

The gameplay uses the standard rules of baseball, and includes a brief guide to the basics in the instructions, but is unlike any other game because you're playing in and for the honour of your street.

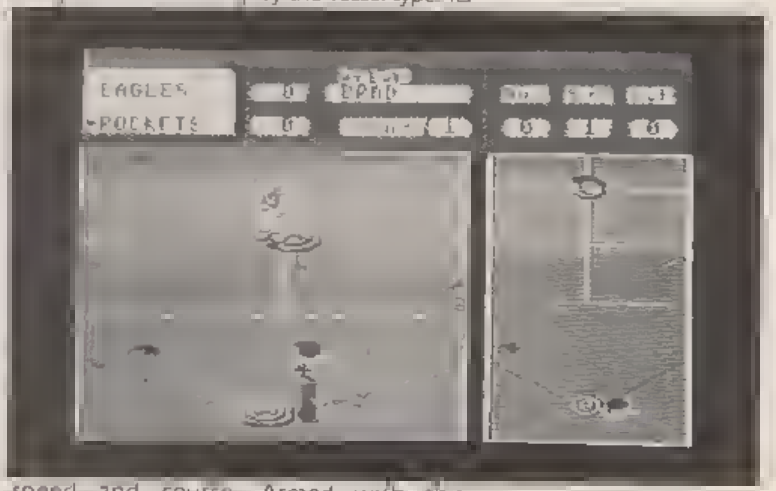
Into battle

Last year Epyx tried to shake its sports games reputation by releasing Destroyer. This put you at the helm of a Fletcher class destroyer and sent you on a variety of missions ranging from a subhunt to convoy escort duties. Now Destroyer has been followed up as Epyx plunges into the depths with Sub Battle Simulator.

At the helm of either an American or Japanese sub you must complete a series of missions to deliver important supplies or attack a convoy.

Your submarine is armed with 16 forward and 8 aft torpedoes, as well as a deck gun for finishing off crippled ships and an anti-aircraft gun to battle enemy aircraft. The submarine's greatest weapon is surprise since it can sail deep under the sea to surface and strike at the heart of a convoy. The payoff for this is that you can only remain underwater as long as you have air for the crew and power in the batteries for the electric motors. If you run out of either you must surface and recharge. If you plan your mission well you will arrive at your patrol zone fully charged and armed for action.

As you close on the enemy you can use your sonar to track their relative position and then the periscope to activate the target computer. This will display the vessel type, its speed and course. Armed with this information you can plan your attack. Your attack run will vary according to the ships in the enemy convoy. For example, if the enemy consists entirely of unescorted freighters you can simply steam up and take them out with both torpedoes and the deck gun. However, if a destroyer is present you will have to be more subtle in your approach and either attack the convoy on the destroyer's blind side or take on the destroyer first before turning on the rest of the fleet. Aircraft carriers can mean big trouble as they could launch planes if they're given enough warning so you must plan your attack well and be ready to man the anti-aircraft guns or dive to safety.



Street Sports Baseball

Sub Battle Simulator offers commanders the chance to tackle single missions or report for full wartime patrols. The latter end either in failure at the bottom of the sea or in glory as the war ends.

There's more to come

Epyx's line up for 1988 includes Street Sports Basketball, Tank Battle Simulator, Print Magic (a Printshop style program to create cards, banners, letterheads and calendars) and the long awaited sequel to Impossible Mission (Impossible Mission II) which provides more of the same but with improved graphics.

Professional programming

Don't make your program a nightmare to read and correct. If you follow our simple guidelines, they'll not only look good - they may even run faster

By Fin Fahey

Have you ever tried working on a program you 'put down' a couple of months before and haven't looked at since? If so you may, like me, have experienced that sinking feeling that comes from the realisation that you find the whole thing utterly incomprehensible, and haven't a clue what you were thinking about when you wrote it.

It's a hard lesson to learn, but a useful one. Because programs, both Basic and machine code, can be easily laid out in such a way that coming back to them will be like reading an old and familiar book.

Some passing REMarks

First things first. Let's start with Basic, and specifically REM statements. REMs are not there just as a language designer's whim - they're meant to be used. They can tell you what some of those impenetrable bits of code do at a glance.

REMs at the end of code lines help a lot, but actual REM lines, some of them blank, are very useful for breaking code up into chunks. Try reading an article that isn't split into paragraphs! I usually label a piece of code that forms a unit with a REM explanation and blank lines before and after, like this:

```
97 REM
98 REM PRINT AT X,Y
99 REM
100 PRINT [HOME],TAB(X),LEFT$(ZD$,Y);X$,
110 RETURN
120 REM    etc
```

In passing, notice that the functional line numbers go up in tens, leaving room for new lines to be inserted. Some of you may thing that this may be too obvious to point out, but having seen a lot of program submissions, I've noticed that many people number their lines consecutively. Honestly, you don't really want to be renumbering the program every five minutes.

Next, don't be afraid of spaces, unless you're writing a very large program and may run out of space (in which case you could use a Basic compressor to remove all the REMs and spaces). Surely

```
3900 IF SP<>OP OR SG<>OG OR SC<>OC THEN 3950. REM NO SIMILARITIES FOUND
```

is a lot easier to understand than

```
3900 IFSP<>OPORSG<>OGORSC OCTHEN3950
```

Just by the way - don't try to do too much on one line either. Long lines of code usually win the prize for incomprehensibility.

Structured programming

We've seen how to make programs a bit prettier, now let's go deeper and see how a piece of code can be structured to best effect.

Program structure is a phrase that means many things to many people. In the commercial world it can be a cruelly rigorous procedure, since commercial code must be easily testable and alterable. All extra time spent on programming eats into the profits.

We, however, don't need to be too fanatical about structure. A little goes a long way.

Flowcharting

Once you have decided what a program is going to do, it may help to draw a flowchart. This identifies the main flow of control through the program and can describe the main loops and decisions to be made therein. Figure 1 shows a flowchart for a simple arcade game.

Many programs follow this structure - they have a program initialisation, and a program end, which usually just ENDs the program. In between is a game loop which itself may have a beginning and end. Between these is the game itself. The flowchart will translate directly into a program:

```
10 REM ********************
20 REM
30 REM SOME GAME OR OTHER
40 REM BY A HACKER
50 REM
60 REM ********************
70 REM
497 REM
498 REM CONTROL SECTION
499 REM
500 GOSUB 10000 REM INITIALISE PROGRAM
510 GOSUB 5000 REM INITIALISE GAME
520 GOSUB 1000 REM PLAY ONE GAME
530 GOSUB 7000 REM END GAME
540 IF X$="Y" THEN 510 REM NEXT GAME
550 GOSUB 9000 REM END PROGRAM
997 REM
998 REM PLAY ONE GAME
999 REM
1000 RETURN    etc
```

and there we are, the program's written.

Note the liberal use of REMs. What's that you say - it doesn't do anything! Of course it does - once the internal functions of all the subroutines in it have been identified and coded. The point is that structuring the program in this way makes it a lot more testable and readable - you can plug the routines in one at a time and test them in order rather than diving into the code in one block, something known as monolithic programming, which is deeply undesirable.

Useful pointers

A number of pointers emerge from the example above. Program initialisation takes place at line 10000, very near the end of the program. This is because in interpreted Basic, the most frequently used subroutines should be placed near the start of the program, since the interpreter has to scan through the code looking for a line number every time that number is called. The fewer the lines it has to look at before it gets to the target line, the faster the code.

If your program uses DATA statements, always, without exception, put these last of all in the program. Once they've been READ, they're dead weight, nothing more. Other ways of getting program speed up are to remove the REM statements and to cram a lot of things into one line to reduce the number of lines needed to be scanned. As we've already seen, this reduces program readability, so be cautious.

Program initialisation is only called once, so we don't care if it's relatively slow. But if we want to move a sprite, we want some speed. For this reason, room has been left above the Control Section, which starts on line 500, for the most frequently called subroutines of all.

System subroutines

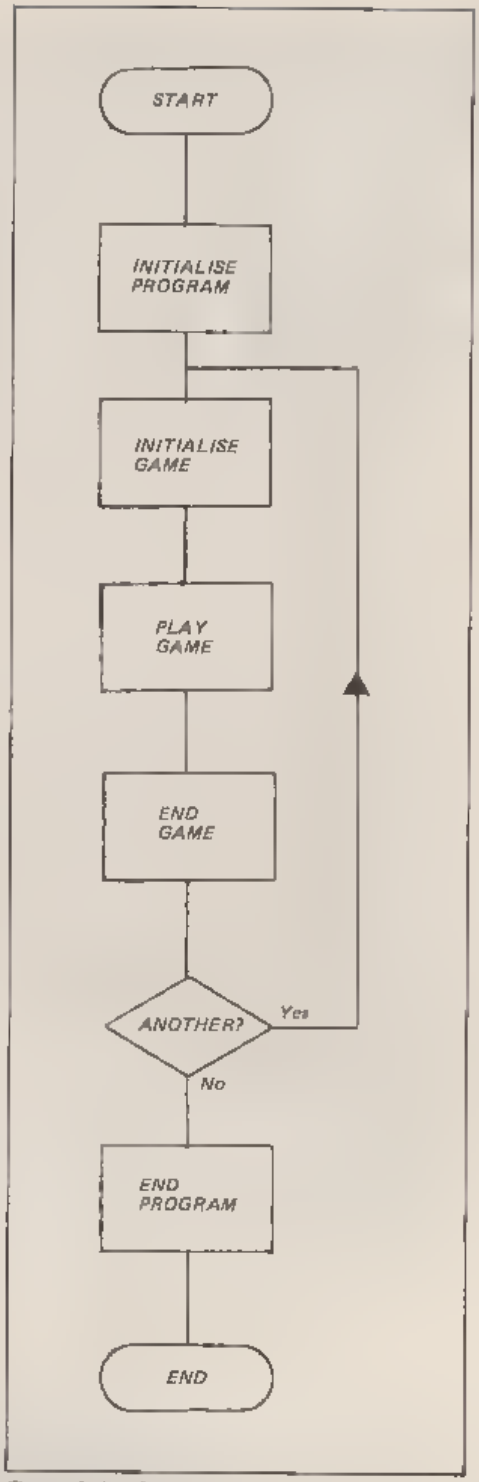
Some of these routines are known as system subroutines. An example would be the PRINT AT routine we showed you earlier. Commodore Basic doesn't have a command to print a string at any point on the screen, so you may need to write one. This may not only be used repeatedly in any one program, but you may wish to use it in many different programs. In the games above, it would be used to print both score and hi-score on the screen.

If you create a library of such sub-routines, it's easy to use this as the basis of your future programs, deleting the ones you don't want.

Looking at our PRINT AT subroutine, we can see that this employs things called parameters. Parameters come in two forms.

Input parameters contain information that the routine needs to perform its function. In the case of the PRINT AT routine, the parameters are X and Y, which contain the X and Y coordinates for the routine, and XS, which is the string to be printed.

Output parameters contain information passed back to the main program by the routine. Our PRINT AT returns no such information, but we can see the process going on within the simple game skeleton above.



One of the functions of sub-routine 7000, besides updating the high-score, is to return a string XI$ which shows whether or not the user wishes to continue. The subroutine might look like this:

```
7000 IF SC<>HS THEN HS=SC REM UPDATE
HIGH SCORE
7010 PRINT [CLR],:X=5 Y=10 XS="DO YOU
WANT ANOTHER GAME (Y/N)?:GOSUB 100
7020 GET XI$ IF XI$ < >"Y" AND XI$ < >"N"
THEN 7020
7030 RETURN
```

Notice that I start all parameters with the letters X or Y. First of all this renders them easily identifiable, secondly if you eschew using these letters for other variables, it will help to avoid using a variable for more than one thing, which is a frequent source of error.

Variables starting with Z, I reserve for constants used by the sub-routines. These are assigned values by the Program Initialise routine, and never change their value throughout the program. In the case of PRINT AT, ZD$ contains a string of 25 down characters, any number of which can be printed to the screen to position the cursor at the appropriate screen row.

Common problems

Certain errors crop up time and time again in Basic programs, so it's worth pointing out a few, although these come under the heading of program content, not style. The commonest error experienced by Your Commodore readers (but not those of you sensible enough to buy CDU) involves getting the dreaded message

OUT OF DATA ERROR

Ninety-nine out of 100 phone messages received in the Your Commodore office are from readers who have typed in a Basic loader and have this problem. The reason for it is that the program is trying to READ more values than there are DATA entries. If you get it, go back and count. This routine would give you the error

```
100 FOR N=1 to 6 READ X$ NEXT N
200 DATA 1,2,3,4,5
```

Another heinous crime is to GOTO out of a FOR ... NEXT loop. Getting out of one of these before it is complete is something that some people find slightly awkward and they may resort to something like

```
100 FOR N=1 TO 100
110 IF X$(N)="END" THEN GOTO 200
120 PRINT X$(N)
130 NEXT N
```

Don't do it. The poor program thinks it's still FOR ... NEXTing and will throw up an error or worse on the next NEXT it hits. Acceptable is

```
110 IF X$(N)="END" THEN N=100 GOTO 130
```

which ends the loop neatly.

A common source of error arises in IF statements. In the statements

```
100 IF X=1 THEN PRINT " ERROR" :X=0
```

the 'X=0' is only performed when the conditions of the IF are satisfied. If you want such a statement to be performed whatever the state of X1, then it'll have to be on a separate line.

A last one to look for is a classic logic problem.

```
200 IF A$="Y" OR A$="N" THEN 400
```

is a reasonable program line but can you see what is wrong with

```
200 IF A$<>"Y" OR A$<>"N" THEN 400
```

Yes - this statement is always true, and therefore pointless. What the writer meant instead of the 'OR' was 'AND'. I'll leave it to you to understand why.

Machine code

Many of the principles we've detailed for Basic also apply to machine code. The main proviso is that this is assembled, not interpreted, so it doesn't matter speed-wise where you put procedures in an assembler program, readability is the thing (even more so!). Assembler remark lines are usually signified by a semi-colon. Use them.

The need to flowchart and to build libraries of parameterised subroutines is paramount in assembly programming. One thing to avoid is endlessly reinventing the wheel. Life's too short. The principle of building easily understandable and testable structures is just the same as in Basic.

A Parthian shot

Finally, a small problem for you. You'll find a program on the disk called BUGGED which, unlike all the others, contains a number of bugs. Your job is to remove all the bugs (I'm not telling you how many there are), plus as far as you can render the program fast and readable. To make matters worse, I'm not even going to tell you what it's supposed to do. The best entry gets a free CDU binder and 10 free disks. Listings and disks please - we don't have the time to list them ourselves. Good luck!

2 issues FREE

When you take out a subscription to any of these magazines

	EUROPE	MIDDLE EAST	FAR EAST	REST OF THE WORLD
A&B Computing	£27.30	£27.60	£31.10	£28.30
Aeromodeller	£27.00	£27.20	£29.00	£27.50
Citizens Band	£20.10	£20.30	£21.80	£20.50
Clocks	£32.40	£32.60	£34.70	£33.00
Commodore Disk User	£18.00	£18.20	£19.30	£18.40
Disk User	£21.60	£21.70	£23.20	£22.00
Electronics Digest	£12.90	£13.00	£13.70	£13.10
Electronics Today International	£21.00	£21.20	£22.80	£21.50
Ham Radio Today	£20.10	£20.30	£22.00	£20.60
Military Modelling	£22.40	£22.60	£25.20	£23.10
Model Boats	£20.00	£20.10	£21.80	£20.40
Model Engineer	£35.40	£35.70	£39.00	£36.30
Photography	£22.00	£22.30	£25.00	£22.80
Photoplay	£16.70	£16.90	£18.70	£17.20
Popular Crafts	£21.40	£21.60	£23.60	£22.00
Radio Control Model Cars	£21.10	£21.30	£23.10	£21.60
Radio Control Boat Modeller	£10.00	£10.20	£11.10	£10.30
RCM&E	£20.40	£20.60	£22.90	£21.00
Radio Control Scale Aircraft Quarterly	£11.10	£11.20	£12.00	£11.30
Radio Modeller	£20.00	£20.20	£22.40	£20.60
Sea Classic International	£10.70	£10.80	£12.00	£11.00
Scale Models International	£19.60	£19.80	£21.50	£20.10
Video Today	£20.20	£20.40	£22.30	£20.70
Which Video?	£19.40	£19.50	£21.20	£19.80
Woodworker	£22.80	£23.00	£25.70	£23.50
Your Commodore	£22.00	£22.30	£25.20	£22.80
Model Railways	£19.10	£19.30	£21.50	£19.70
Practical Wargamer	£7.10	£7.20	£8.00	£7.40

The Above Offer Applies to Overseas Subscriptions Only

This offer is also open to existing subscribers who may wish to extend their current subscriptions.

Please commence my subscription(s) To Issue
I enclose my cheque/money order for £ with the
or debit £ made payable to Argus Specialist Publications Ltd.
valid from to from my Access/Barclaycard No.
Name Signature
Address

Send this form with your remittance to:

INFONET LTD. (S.O. 88) 5 River Park Estate, Berkhamsted, Herts. HP4 1HL.

DON'T GET LEFT OUT... GET IN ON THE ACTION

COMMODORE DISK USER is a lot more than just another computer magazine. Every issue carries a diskette containing more than £30 worth of software ranging from serious programming utilities to arcade games. There are plenty of Commodore magazines on the market, but we believe that this is the *first* to cater for disk users of all ages and tastes.

COMMODORE DISK USER is what you have been waiting for – take out a subscription TODAY!

Please begin my subscription(s) to COMMODORE DISK USER with the ..... issue
I enclose my cheque/money order for £ ..... made payable to Argus Specialist Publications Ltd
or debit £ ..... from my Access/Barclaycard No .....
valid from ..... to .....
NAME (Mr/Mrs/Miss) .....
ADDRESS .....
.....
Postcode .....
Signature .....
Date .....

Please use BLOCK CAPITALS and use post codes

SUBSCRIPTION RATES

£15.00 for 6 issues U.K.
£18.00 for 6 issues EUROPE
£18.20 for 6 issues MIDDLE EAST
£19.30 for 6 issues FAR EAST
£18.40 for 6 issues REST OF WORLD
Airmail Subscription Rates on Request

Send your remittance to:
INFONET LTD., 5 River Park Estate, Berkhamsted, Herts. HP4 1HL.